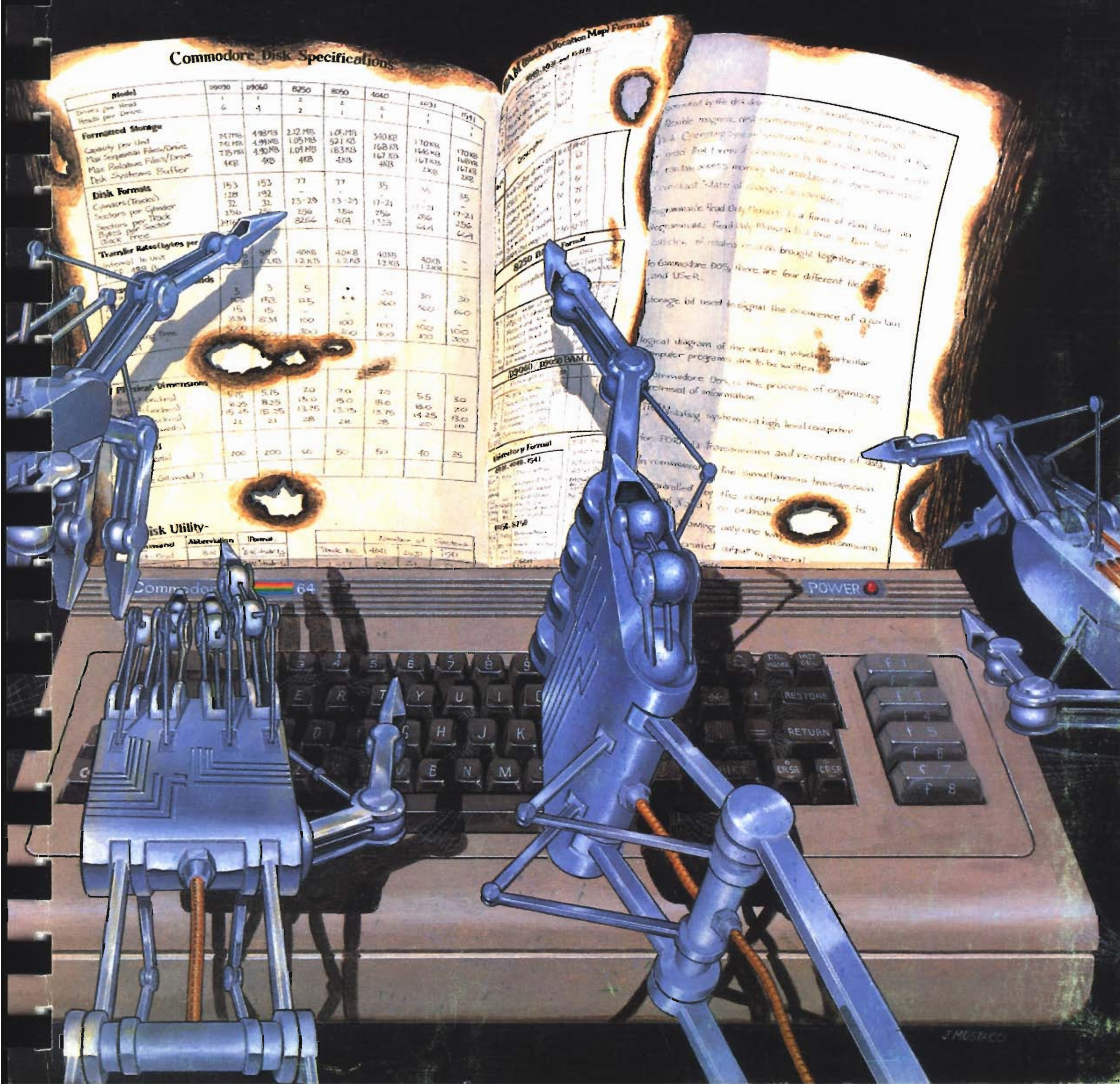
The Complete Commodore Inner Space Anthology

Karl J.H. Hildon



The Complete Commodore Inner Space Anthology

Karl J.H. Hildon

The Making Of. . .

What you see before you is the collection, culmination, and collation of almost 5 years of information about Commodore Computers. It all began with The Best of The Transactor Volume 2 and a photocopier with a reduction feature. It occurred to me that if all my most referenced facts were together on one page they would be infinitely more useful. Memory maps, conversion charts, machine code tables, and everything else went into the copier over and over until they were small enough to paste together on one sheet. But the photocopier had its drawbacks; each new reduction meant a drop in quality and the distortion factor of the copier had the top lines slanting down and the bottom lines slanting up.

After I departed from Commodore to run The Transactor independently, I was thrust into the world of the phototypesetter, the ultimate printer. At first I was totally consumed by the superb quality of the type, but that didn't last long. I began experimenting with point sizes (character size), leading (line spacing), and the over 300 other commands that are available including an entire text programming language. With vertical spacing down to $1/576$th of an inch and horizontal accuracy to $1/1296$th of an inch, I found myself accounting for every fraction. This exact science of typesetting was the perfect answer to the question of how the next generation of compact reference material would be created.

After about eight months of practice I decided it was time. Four months later The Special Reference Issue of The Transactor (Volume 4, Issue 5) was released. The brown cover earned it the nickname, "The Brown Bible" and it wasn't long before many were referring to it as "the most photocopied magazine of all time". Everyone seemed to be happy with it, except me.

It was about six months later when Attic Typesetting took delivery of the first Quadex Preview in Canada, a fabulous device that shows on a screen exactly what the type machine will produce. Typesetting: the Science, became Typesetting: the Art. It was then I decided the next generation was within my reach. Although the Preview simplified the task by easily ten-fold, the amount of target material had more than tripled. After eight months of organizing (in the time between making magazines) and almost two months of double shifts at the type shop, I now find myself writing this paragraph. The Complete Commodore Inner Space Anthology is finally finished.

Acknowledgements

Special thanks to Richard T. Evers and Chris J. Zamara: two very special talents inside two very special individuals. Invaluable assistance lacked a true definition until you guys.

Extra special thanks to Jim Butterfield: Jim was responsible for the memory maps of all the computers, each one a masterpiece of information dissemination. The original idea of the SuperChart was also Jim's. Your influence and inspiration are exceeded only by your generosity, three quantities I could only hope my appreciation might one day equal.

Attic Typesetting, namely Phyllis Fast and Nate Redmon: your patience and understanding are outweighed only by your typesetting equipment.

Special thanks to Bill Maclean: for backing me up, all the way.

Others I wish to thank include Len Lindsay for providing COMAL memory maps and other valuable data; Jim Gracely of Commodore for providing the Computer Club listing; Nick Sullivan, Editor of TPUG Magazine, for necessary data to create the Chord Derivatives; David Berezowski for finding me a MOS Data Catalog; Domenic DeFrancesco for his help with hardware problems; Jim Yost, Louis Sander, and Colin Arneld for sending in their notes that allowed for improvements; and Raeto Collin West for setting the standard with Programming the PET/CBM.

Cover Design by John Mostacci **Printed in Canada** **ISBN 0-9692086-0-X**

Dedicated to John A. Hildon, my dad.

The Complete Commodore Inner Space Anthology

SuperCharts

BASIC Section

COMAL Section

Printer Section

Business Software Section

Machine Language Section

Memory Maps

Disk Drives Section

Music Section

Video Section

Telecomputing Section

Hardware Section

Arithmetic and Mathematics

BASIC – Beginners All–Purpose Symbolic Instruction Code

Commands and Statements

Command/ Statement	Example	Purpose
CLOSE	10 CLOSE n	Closes logical file 'n'.
CLR	CLR	Sets variables to zero or null.
CMD	CMD D	Keep ieee device 'D' open to monitor bus.
CONT	CONT	Continue program execution after a stop command. No program changes are permitted.
DATA	10 DATA 1,2,3,4 20 DATA TOM, SUE 30 DATA " DOE, TOM "	Specifies data to be read left to right. Alphabetics do not need to be enclosed in quotes. if strings contain spaces, commas, colons, or graphic characters, the string must be enclosed in quotes.
DEF	10 DEF FN R(X)	Defines function 'R'
DIM	10 DIM A(n) 20 DIM A(n,m,o,p) 30 DIM A(n),B(m) 40 DIM A(N) 50 DIM A$(n)	Specifies maximum number of elements in an array or matrix. Specifies maximum number of dimensions in an array. Number of arrays limited by memory. May be dimensioned dynamically. Strings to be dimensioned.
END	999 END	Terminates program execution.
FOR	10 FOR I = 1 TO 10	Begins repetitive loop, specifying loop variable and number of intended iterations (in this example 'I' for 10 iterations).
FRE	PRINT FRE (0)	Returns number of bytes of available memory.
GET	10 GET C 20 GET C$ 30 GET #d, C 40 GET #d, C$	Accepts single numeric character from keyboard. Accepts single string character from keyboard. Accepts single character from specified logical file. Accepts specified single string character from logical file.
GOSUB	10 GOSUB n	Begins execution of a subroutine which begins on line 'n'.
GOTO	10 GOTO n	Transfer program execution to line n.
IF...GOTO	10 IF X = 10 GOTO n	Transfers execution to line 'n' if result of condition is true.
IF...THEN	10 IF X = 10 THEN Y = 3	Code following THEN is executed only if result of condition is true. May also be followed by line number to transfer execution.
INPUT	10 INPUT A 20 INPUT A$ 30 INPUT A,A$,B,B$ 40 INPUT #d, A 50 INPUT #d, a$ 60 INPUT #d, A,A$,B,B$	Accepts value of 'A' from keyboard. Accepts value of string variable 'A' from keyboard. The string does not have to be enclosed in quotes. Accepts specified values from keyboard. Accepts value of 'A' from logical file 'd'. accepts specified string from logical file 'd'. Accepts specified values and string from logical file 'd'. Strings do not have to be enclosed in quotes.
LET	LET X = 10	Optional. Assigns variable 'X' the value of 10.
LIST	LIST LIST -n LIST n-m LIST n-	Lists current program. Lists current program through line 'n'. Lists lines 'n' through 'm' of current program. Lists current program from line 'n' to end.
LOAD	10 LOAD 20 LOAD " NAME " 30 LOAD " NAME ", d 30 LOAD " NAME ", d, c	Loads next encountered program from tape unit into memory. Loads program or file 'NAME' into memory from tape unit. Loads specified file 'NAME' from device 'd'. Loads specified file 'NAME' from device 'd' for command 'c'. (VIC/C64 only – c = 1 for direct memory load)
NEW	NEW	Deletes current program in memory, sets variables to zero.
NEXT	NEXT	Indicates end of code contained in a FOR/NEXT loop.
ON...GOSUB	10 ON A GOSUB l, m, n	Begins execution of subroutine which begins on specified line (in this example, 'l','m', or 'n') depending on value of index 'A'.
ON...GOTO	10 ON A GOTO l, m, n	Transfers control to specified line 'l','m', or 'n' depending on value of index 'A'.
OPEN	10 OPEN a 20 OPEN a, d 30 OPEN a, d, c 40 OPEN a, d, c, " NAME "	Opens logical file 'a' for read only from tape unit. Opens logical file 'a' for read only from device 'd'. Opens logical file 'a' for command 'c' from device 'd'. Opens logical file 'a' on device 'd'. If device 'd' accepts formatted files, file name is positioned for command.
PEEK	PEEK(a) PEEK(A)	Returns byte value from address 'a'. Address can be dynamic.
POKE	POKE a, b POKE A, B	Puts byte 'b' into address 'a'. Parameters can be dynamic.
POS	10 PRINT POS(0)	Prints next available print position (position of cursor on screen).
PRINT	10 PRINT A 20 PRINT A$ 30 PRINT A, A$ 40 PRINT #d, A 50 PRINT #d, A$	Prints value 'A' on display screen. Prints specified string on screen. Prints specified values or strings on screen, beginning in next available print position (pre-tabbed positions are in columns 10,20,30,40 etc.). Prints value of 'A' on device 'd'. Prints specified string on device 'd'.
READ	10 READ A$, B$	Reads next two data elements into variables A$ and B$.
REM	10 REM Comment	Remark indicator. Execution skips entire line.
RESTORE	10 RESTORE	Resets data pointer so that next READ receives first element of first DATA statement.

Commands and Statements, cont'd

Command/ Statement	Example	Purpose
RETURN	9990 RETURN	Subroutine exit; transfers control to the statement following most recent gosub directing transfer to the subroutine.
RUN	RUN RUN n	Begins execution of program at lowest line number. Begins execution of program a line 'n'.
SAVE	SAVE "NAME" SAVE "NAME", d SAVE "NAME", d, c	Saves current file or program 'NAME' on tape unit. Saves current program or file 'NAME' on device 'd'. Saves file 'NAME' on device 'd'. 'c' specifies eof or eot.
STEP	10 FOR I = 1 TO 10 STEP 2	Alters loop variable increment.
STOP	STOP	Stops program execution.
SYS	SYS (x)	Complete control is transferred to a machine language program at the decimal address contained in the argument. Brackets optional.
USR	USR (x)	Transfers program control to a program whose address is at locations 1 and 2 (VIC/C64 - locations 784,785). 'x' is a parameter passed to and from the machine language program.
VERIFY	VERIFY VERIFY "NAME" VERIFY "NAME", d	Verifies current program against next program on tape unit. Verifies current program against program 'NAME' on tape unit. Verifies current program 'NAME' on device 'd'.
WAIT	WAIT a, b, c	Halts execution of Basic until contents of address 'a', and'ed with value 'b' and exclusive or'ed with value 'c', is not equal to zero. 'c' is optional and defaults to zero.

String Functions

Function	Example	Purpose
ASC	10 A = ASC("XYZ")	Returns the integer value corresponding to ASCII code of the first character in string.
CHR$	10 A$ = CHR$(n)	Returns character corresponding to ASCII code number.
LEFT$	10 PRINT LEFT$(X$, a)	Returns leftmost 'a' characters from string.
LEN	10 PRINT LEN(X$)	Returns length of string.
MID$	10 PRINT MID$(X$, a, b)	Returns 'b' characters from string, starting with the 'a'th character.
RIGHT$	10 PRINT RIGHT$(X$, a)	Returns rightmost 'a' characters from string.
STR$	10 A$ = STR$(A)	Returns string representation of variable 'A'
VAL	10 A = VAL(A$) 20 A = VAL("A")	Returns numeric representation of string. If string not numeric, returns "0".

ASC, LEN and VAL functions return numeric results. They may be used as part of any numerical expression.
Assignment statements are used here for examples only; other statement types may be used.

Arithmetic Functions

Function	Example	Purpose
ABS	10 C = ABS(A)	Returns magnitude of argument without regard to sign.
ATN	10 C = ATN(A)	Returns arctangent of argument. 'c' will be expressed in radians.
COS	10 C = COS(A)	Returns cosine of argument. 'A' must be expressed in radians.
DEF FN	10 DEF FNA(B) = C*D	Allows user to define a function. Function label 'a' must be a single letter; argument 'b' is a dummy.
EXP	10 C = EXP(A)	Returns constant 'e' raised to the power of the argument.
INT	10 C = INT(A)	Returns largest integer less than or equal to argument.
LOG	10 C = LOG(A)	Returns natural logarithm of argument. Argument must be greater than or equal to zero.
RND	10 C = RND(A)	Generates a random number between zero and one. If 'a' is less then 0, the same random number is produced in each call to rnd. If 'a' = 0, the same sequence of random number is generated each time rnd is called. If 'a' is greater than 0, a new sequence is produced for each call to rnd.
SGN	10 C = SGN(A)	Returns -1 if argument is negative, returns 0 if argument is zero, and returns + 1 if argument is positive.
SIN	10 C = SIN(A)	Returns sin of argument. 'A' must be expressed in radians.
SQR	10 C = SQR(A)	Returns the square root of argument.
TAN	10 C = TAN(A)	Returns tangent of argument. 'A' must be expressed in radians.

Arithmetic Operators

Symbol	Example	Purpose
=	10 A = B 20 LET A = B	Assigns a value to a variable. LET is optional.
↑	30 PRINT A↑2	Exponentiation
/	40 C = A/8	Division.
*	50 C = A*8	Multiplication.
+	60 C = A + 8	Addition.
-	70 C = A-8	Subtraction.
=	10 IF A = B THEN PRINT C	'A' Equals 'B'.
<>	10 IF A<>B THEN C = 4	'A' Does not equal 'b'.
<	10 IF A<B THEN C$ = "X"	'A' Is less than 'B'.
>	10 IF A>B THEN C$ = "Y"	'A' Is greater than 'B'.
< =	10 IF A< = B THEN C = 20	'A' Is less than or equal to 'B'.
> =	10 IF A> = B THEN C = D-1	'A' Is greater than or equal to 'B'.
AND	10 IF A AND B THEN C = 9	'A' and 'B' must both be true for statement 10 to be true.
OR	20 IF A OR B THEN C = 9	'A' must be true or 'B' must be true for statement 20 to be true.
NOT	30 IF NOT A THEN PRINT C	Expression is true if 'A' is false.

Note: the numerical values used in the evaluation of logical comparisons are: 'true' is any non-zero number and 'false' is zero.

Hierarchy of Operations

Operator	Description
( )	Brackets always dictate priority
↑	Exponentiation
-	Negation (unary minus)
* /	Multiplication & Division
+ -	Addition & Subtraction
< = >	Relational Operations
NOT	Logical NOT (Integer two's complement)
AND	Logical AND
OR	Logical OR

Reserved Variables

Variable	Purpose
DS	Disk Status number (except 2.0)
DS$	Disk Status string (except 2.0)
EL	Error Line (B Series/ + 4/C16 only)
ER	Error number (B Series/ + 4/C16 only)
ERR$(	Error String array. See table for messages. (B Series/ + 4/C16 only)
TI	Time in Jiffies (1/60th's sec.) since power up or TI$ reset (except B Series)
TI$	Time in HHMMSS
ST	The Status variable. See table for functions.

Special Symbols

Symbols	Example	Purpose
:	10 A = 1:B = 2:C = 3	Allows multiple statements on a line.
;	10 PRINT A;B 20 PRINT A$;B$	Suppress Carriage Return for same line printing. Optional after $ or % variables.
.	X = 10.99	Decimal Point
,	10 PRINT A, B LOAD "NAME",d	Allows same line printing. Elements are separated and printed in pre-'tab'ed print positions (columns 10,20,30, etc.). Separates parameters in load, save, open, mid$, on..goto, etc.
?	10 ?A	Abbreviation for 'print'. Stores as one character; lists as word PRINT.
$	10 A$ = "ABCDEFG"	String identifier.
%	10 A% = INT(X)	Integer identifier.
"	10 A$ = "ABCDEFG"	String enclosures.
π	10 C = π*D	Value of Pi 3.1415927.

Basic 4.0 Disk Commands

Function	Example	Purpose
APPEND	10 APPEND#d, "NAME"	Open file 'NAME' on device 'd' for appending. New data is added to end of existing data.
BACKUP	BACKUP D0 TO D1	Duplicate disk in drive 0 onto disk in drive 1
CATALOG	CATALOG D0	Displays list of filenames in specified drive.
COLLECT	COLLECT D1	Purges disk in specified drive of any improperly closed files (indicated by * beside file type).
CONCAT	CONCAT "NAME1" TO "NAME2", D1	Concatenates file "NAME1" to file "NAME2". Ie. NAME2 = NAME2 + NAME1
COPY	COPY "NAME",D0 TO "NAME",D1 COPY "NAME",D0 TO "DUP",D0 COPY D0 TO D1	Copies file "NAME" from drive 0 to drive 1 Makes duplicate of file "NAME" Copies entire contents from D0 to D1
DCLOSE	DCLOSE #n	Closes disk logical file 'n'
DIRECTORY	DIRECTORY D0	Exact same as Catalog. Use preference.
DLOAD	DLOAD "NAME",Dd,Uu	Loads program "NAME" from drive 'd' on unit 'u'
DOPEN	DOPEN#n, "NAME",Dd,Uu DOPEN#n, "NAME",Dd,Uu,W	Opens file "NAME" for reading from drive 'd', unit 'u'. Default values: d = 0, u = 8. Data is retrieved through file number 'n'. Opens file "NAME" for writing to drive 'd', unit 'u'. Not necessary for RELative files.
DSAVE	DSAVE "NAME",Dd,Uu	Saves current program to drive 'd' on unit 'u' as file "NAME"
HEADER	HEADER "DISKNAME",Dd,Iid,Uu	Formats disk in drive 'd' unit 'u' assigning it a "DISKNAME" and 'id'.
RECORD	10 RECORD#n, a	Positions relative file open on logical file number 'n' to record number 'a'. 'a' may be dynamic but must be enclosed in brackets.
RENAME	RENAME "NAME" TO "NEWNAME",D0	Changes a file name.
SCRATCH	SCRATCH "NAME",D1	Eliminates file "NAME" from disk.

BASIC RAM Memory Allocation

BASIC Text	Variable Table	Arrays Space	Empty Space	String Space	
0 000					

	Start of BASIC	Start of Variables	Start of Arrays	End of Arrays	Bottom of Strings	Top of Memory
BASIC 4/2:	$28,29	$2A,2B	$2C,2D	$2E,2F	$30,31	$34,35
VIC/C64:	$2B,2C	$2D,2E	$2F,30	$31,32	$33,34	$37,38
B Series:	$2D,2E	$31,32	$35,36	$37,38	$3B,3C	$0380,0381
+4/C16:	$2B,2C	$2D,2E	$2F,30	$31,32	$33,34	$37,38

BASIC Text Line Structure

0	L	H	L	H	line text	0	L	H	L	H	line text	0	0	0

Start of BASIC; Link (L H) — pointer to next line; line # (L H); Link (L H) — pointer to next line; line # (L H); end of prg (0 0 0)

'FOR' Stack Entry

LO	Pointer to first
HI	statement in loop
HI	Line number of first
LO	statement in loop
M4	
M3	
M2	'TO' value
M1	
EXP	
	Sign of 'STEP'
M4	
M3	
M2	'STEP' value
M1	
EXP	
HI	Pointer to
LO	'FOR' variable
$81	'FOR' Token (LAST ON)

Variable Formats

Floating Point

N	N					

name (NN); exponent + 128; msb — lsb

Integer

J	J	H	L	0	0	0

name (JJ%); value; unused

String

S	G		L	H	0	0

name (SG$); length of string in bytes; start address of string

'GOSUB' Stack Entry

HI	Pointer to
LO	'GOSUB' statement
HI	Line Number of
LO	'GOSUB' statement
$8D	'GOSUB' Token (LAST ON)

Reserved System Variables

ST - The Status Variable

Bit	Val	Cassette Read	IEEE/Serial	Tape Load/Ver.	Vic/64 RS-232
0-7	0	OK	OK	OK	OK
0	1		time out on write		parity error
1	2		time out on read		framing error
2	4	short block		short block	rec. buffer overrun
3	8	long block		long block	unused
4	16	unrecoverable read error		any mismatch	CTS signal missing
5	32	checksum error		checksum error	unused
6	64	end of file	EOI		DSR signal missing
7	-128	end of tape	device not present	end of tape	break detected

DS & DS$ - Disk Status Variables

DS	Error Description
0	OK, no error exists
1	files scratched response (not an error)
2-19	Unused: can occur, should be ignored
20	read error; block header not found
21	read error; sync character not found
22	read error; data block not present
23	read error; checksum error in data
24	read error; byte decoding error
25	write error; write verify error
26	write protect on
27	read error; checksum error in header
28	write error; data extends into next block
29	disk id mismatch
30	syntax error; general syntax
31	syntax error; invalid command
32	syntax error; command line > 58 chars
33	syntax error; invalid filename
34	syntax error; no filename given
39	syntax error; command file not given
50	record not present
51	overflow in record
52	file too large
60	file open for write
61	file not open
62	file not found
63	file exists
64	file type mismatch
65	no block; t,s is next available block
66	illegal track or sector
67	illegal system track or sector
70	no channels (available)
71	dir error (directory error)
72	disk full or directory full
73	cbm dos v2 (or v2.x for later dos's); power up message, also indicates write attempt with dos mismatch
74	drive not ready
75	format speed error
76	controller error

Additional B Series Commands

Function	Example	Purpose
BANK	BANK b	Sets bank number to 'b'.
BLOAD	BLOAD "NAME",Dd,Uu,ON Bb,Pp	Loads file "NAME" from drive 'd' unit 'u' into bank 'b' at position 'p'
BSAVE	BSAVE "NAME" ON Bb,Pp1 to Pp2	Saves current memory in bank 'b' from address 'p1' to 'p2' as file "NAME" to drive 0 unit 8. Addresses are in decimal.
DCLEAR	DCLEAR D1	Initialize disk in drive 1
DELETE	DELETE 10-30	Deletes lines from current program. Specify line range same as LIST.
DISPOSE	DISPOSE GOSUB	Purges stack of unwanted return addresses (like 'POP')
ELSE	IF ST THEN E = 1 ELSE E = 0	Alternate condition following IF..THEN. May also be used to transfer execution
INSTR	PRINT INSTR (A$, B$)	Returns position of string B$ within A$. Returns 0 if not found.
KEY	KEY KEYn, "CATALOG D0" + CHR$(13)	Displays list of function key definitions Defines function key 'n'.
PUDEF	PUDEF "-.,£"	Re-defines Print Using format characters. Default is " ,.$". In this example, space is changed to '-', comma to period, period to comma, and dollars to pounds.
RESUME	RESUME RESUME n RESUME NEXT	Continues execution after program error or editing Resumes execution at line 'n' Resumes execution at start of current active FOR/NEXT
TRAP	TRAP 50000	Specifies routine at line 50000 as an ON ERROR routine.
USING	PRINT USING "-$##,###";X	Specifies format to be used for numerical output.

Additional +4, C16 Commands

Function	Example	Purpose
		Editing:
AUTO	AUTO 100, 10	Supply line numbers starting with 100 in increments of 10
DELETE	DELETE -10	Delete BASIC lines up to line 10. Parameters work like LIST.
HELP	HELP	Hi-lites BASIC execution error in RVS field
KEY	KEY KEY FK, FK$	Display Function Key assignments Define Function Key FK (1-8) as FK$. Allows any string expression.
RENUMBER	RENUMBER 1000, 10, 500	Renumber BASIC text starting with line 1000 in increments of 10, from line 500 on.
TROFF	TROFF	Turns BASIC execution trace feature OFF.
TRON	TRON	Turns BASIC execution trace feature ON.
		Structure:
DO LOOP		can be followed by WHILE or UNTIL
EL	PRINT EL	Reserved variable: Error Line
ER	PRINT ER	Reserved variable: Error Number
ERR$	PRINT ERR$(ER)	Reserved variable: Error Message (example would print last error string)
GETKEY	10 GETKEY A$	Instead of 10 GET A$: IF A$ = "" THEN 10
IF THEN ELSE	1000 IF J = K THEN 1010 ELSE STOP	Must all be on same line.
INSTR	INSTR A$, B$, PO	Insert A$ into B$ at position PO.
PRINT USING	PRINT USING F$, A$	Print A$ using format F$
PUDEF	PUDEF "-.,£"	Re-Define USING format characters
RESUME	RESUME 1200	Resume loop at 1200
TRAP	5 TRAP 1000	Equivalent to ON ERROR GOTO 1000
EXIT	2090 EXIT	Terminate loops started with DO
		Graphics
FLASH	100 FLASH A$	Sets flashing attribute on string A$
BOX	BOX CS, X1, Y1, X2, Y2, AN, 1	Draws a box from X1,Y1 to X2,Y2, at an angle AN, filled in with same colour as colour source CS
CHAR	210 CHAR CS, X, Y, A$, 1	Will print A$ at X,Y position on the Hi-Res screen, using colour source CS, reversed.
CIRCLE	CIRCLE 2, X, Y, XR, YR, S, E, A, I	Draws a circle where: 2 = Use Multicolor 1; X,Y = Position of center; XR = X Radius; YR = Y Radius; S = Starting Arc (default 0 degrees); E = Ending Arc (default 360 degrees); A = Clockwise rotation (default 0); I = Increment or Coarseness (default 2)
COLOR	COLOR BK, FG, M1, M2, BD	Set colours for Background, Foreground, Multi-Colour 1, Multi-Colour 2, Border (range 0-15).
DRAW	230 DRAW 4,X1,Y1,X2,Y2,C	Will draw a line from X1,Y1 to X2,Y2 in Border colour

Additional +4, C16 Commands, cont'd

Function	Example	Purpose
GRAPHIC	GRAPHIC M, C	Specify screen mode M. 0 = Text 1 = Multi-Colour Graphic 2 = Hi-Res Graphic 3 = Split-Screen (Text on bottom 3 lines) C <> 0 clears screen.
GRAPHIC CLR	GRAPHIC CLR	Clear current GRAPHIC screen
GSHAPE	250 GSHAPE S$, X1, Y1, M	Gets a shape from S$ and print it on the Hi-Res screen at X1,Y1 using mode M. 0 = Draw Shape as is (default) 1 = Draw Shape inverted 2 = Draw Shape OR'd with Screen 3 = Draw Shape AND'd with Screen 4 = Draw Shape XOR'd with Screen
JOY	PRINT JOY(JS)	Returns direction (0-8) of Joystick 1 or 2 (0-1). Fire Button adds 128 to direction value.
LOCATE	220 LOCATE X1, Y1	Set initial co-ordinates for plotting type commands to X1,Y1
PAINT	PAINT C, X, Y, M	Fills the area surrounding X,Y in colour C using mode M. 0 = Bordered by same colour as C 1 = Bordered by any foreground colour
RCLR	PRINT RCLR (CS)	Returns Colour Source information for: 0 = Background colour number 1 = Foreground colour number 2 = Multi-Colour 1 colour number 3 = Multi-Colour 2 colour number 4 = Border colour number
RDOT	PRINT RDOT (M)	Returns information for the next pixel to be plotted using mode M. 0 = X co-ordinate 1 = Y co-ordinate 2 = Colour Source
RGR	PRINT RGR (0)	Returns current GRAPHIC mode (0-3)
RLUM	PRINT RLUM (CS)	Returns luminance for colour source CS.
SCALE	200 SCALE X	Set scale to: 0 = Standard co-ordinates based on GRAPHIC mode. 1 = 0-1023 co-ordinate system.
SCNCLR	200 SCNCLR	Clears screen in any GRAPHIC mode
SOUND	260 SOUND	Single voice, followed by parameters for note, tone, etc.
SSHAPE	250 SSHAPE S$, X1, Y1, X2, Y2	Saves a shape into S$ from X2,Y2 to X1,Y1 (the diagonally opposite corner)
VOL	270 VOL V	Sets volume from 0 to 8 maximum
		Machine Language:
DEC	DEC "FFFF"	Converts the string FFFF to decimal. Variable can also be used.
HEX$	HEX$(1024)	Converts the number 1024 to a string representing the hexadecimal equivalent. DEC and HEX$ complement much like ASC and CHR$
MONITOR	MONITOR	Enters Machine Language Monitor
F	F EA 6000 7000	Fill memory from ADDR1 to ADDR2 with specified hex value
H	H 6000, 7000, A9 FF	Hunt memory from ADDR1 to ADDR2 for the sequence A9 FF
A	A JSR $FFD2	Assemble, works like Supermon assembler
D	D 6000	Disassemble from $6000 on.
M	M 6000 6050	Memory dump displays memory contents in hex and screen POKE characters.
G	G 6000	Go to $6000 and execute machine language there.
X	X	Exit MLM
S	S "program",08,6000,7000	Save ML program between $6000 and $7000 on device 8
L	L "program"	Load specified program. Load address is contained in file.
R	R	Display registers

B Series / +4 / C16 ESCAPE Key Functions

ESCape +	Function
A	Automatic Insert Mode
B	Set Bottom of Screen Window
C	Cancel Automatic Insert Mode
D	Delete line
E	Use Nonflashing Cursor (B Series only)
F	Use Flashing Cursor (B Series only)
G	Enable Bell
H	Disable Bell
I	Insert a line
J	Move Cursor to Start of Current line
K	Move Cursor to End of Current line
L	Enable Scrolling
M	Disable Scrolling

ESCape +	Function
N	Set Normal Screen display size
O	Cancel Insert, Quote, and Reverse Modes
P	Erase Begin
Q	Erase End
R	Set Reduced Screen display size
S	Use Solid Cursor (B Series only)
T	Set Top of Screen Window
U	Use Underscore Cursor (B Series only)
V	Scroll Up
W	Scroll Down
X	Cancel ESCape
Y	Use Normal Character Set (B Series only)
Z	Use Alternate Character Set (B Series only)

Error Messages

Message	Description
BAD DATA	String data was received from an open file, but the program was expecting numeric data.
BAD SUBSCRIPT	The program was trying to reference an element of an array whose number is outside of the range specified in the DIM statement.
CAN'T CONTINUE	The CONT command will not work, either because the program was never 'RUN', there has been an error, or a line has been edited.
DEVICE NOT PRESENT	The required I/O device was not available for an 'OPEN', 'CLOSE', 'CMD', 'PRINT#', 'INPUT#', or 'GET#'.
DIVISION BY ZERO	Division by zero is a mathematical oddity and not allowed.
EXTRA IGNORED	Too many items of data were typed in response to an input statement. Only the first few items were accepted.
FILE NOT FOUND	If you were looking for a file on tape, an 'end-of-tape' marker was found. If you were looking on a disk, no file with that name exists.
FILE NOT OPEN	The file specified in a 'CLOSE', 'CMD', 'PRINT#', 'INPUT#', or 'GET#', must first be 'OPEN'ed.
FILE OPEN	An attempt was made to OPEN a file using the number of an already open file.
FORMULA TOO COMPLEX	The string expression being evaluated should be split into at least two parts for the system to work with, or a formula has too many parentheses.
ILLEGAL DIRECT	The 'INPUT' statement can only be used within a program, and not in direct mode.
ILLEGAL QUANTITY	A number used as the argument of a function or statement is out of the allowable range.
LOAD	A problem has occured during program LOAD, disk or tape
NEXT WITHOUT FOR	This is caused by either incorrectly nesting loops or having a variable name in a 'NEXT' statement that doesn't correspond with one in a 'FOR' statement.
NOT INPUT FILE	An attempt was made to 'INPUT' or 'GET' data from a file which was specified to be for output only.
NOT OUTPUT FILE	An attempt was made to 'PRINT' data to a file which was specified as input only.
OUT OF DATA	A 'READ' statement was executed but there is no data left unread in a 'DATA' statement.
OUT OF MEMORY	There is no more 'ram' available for program or variables. This may also occur when too many 'FOR' loops have been nested, or when there are too many 'GOSUB's in effect.
OVERFLOW	The result of a computation is larger than the largest number allowed, which is 1.70141884e+38.
REDIM'D ARRAY	An array may only be 'DIM'ensioned once. If an array variable is used before that array is 'DIM'd, an automatic 'DIM' operation is performed on that array setting the number of elements to ten, and any subsequent 'DIM's will cause this error.
REDO FROM START	Character data was typed in during an 'INPUT' statement when numeric data was expected. Just re-type the entry so that it is correct, and the program will continue by itself.
RETURN WITHOUT GOSUB	A 'RETURN' statement was encountered, and no 'GOSUB' command has been issued.
STRING TOO LONG	(except 2.0) Maximum string length is 255 characters. This error will also occur if INPUT# receives more than 80 characters without a carriage return (ie. BASIC input buffer is 80 bytes long), or if a disk filename is longer than 16 characters.
SYNTAX	A statement or command is unrecognizable. A missing or extra parenthesis, misspelled keywords, etc.
TYPE MISMATCH	This error occurs when a number is used in place of a string, or vice-versa.
UNDEF'D FUNCTION	A user defined function was referenced, but it has never been defined using the 'DEF FN' statement.
UNDEF'D STATEMENT	An attempt was made to 'GOTO' or 'GOSUB' or 'RUN' a line number that doesn't exist.
VERIFY	The program on tape or disk does not match the program currently in memory.

Notes

B Series, +4, and C16 Error Messages

This list is a summary of error messages that are displayed by PRINTing ERR$(X) where X equals the value down the left column.

X	Message	Explanation
0	?STOP KEY DETECTED	Occurs when doing a KERNAL I/O function and the STOP key is pressed. May occur during LOAD or SAVE (or OPEN, CLOSE, GET#, INPUT#, PRINT# when the cassette tape is moving). CLOSE any open write files to save data.
1	?TOO MANY FILES	Maximum OPEN files is ten.
2	?FILE OPEN	An attempt was made to OPEN or DOPEN a file with a file number already in use.
3	?FILE NOT OPEN	An attempt was made to access a file not previously OPEN or DOPENed
4	?FILE NOT FOUND	The file specified in OPEN or LOAD was not found on the device specified. For tape I/O, an end of tape marker was encountered.
5	?DEVICE NOT PRESENT	An attempt was made to access a device not currently connected or powered-up on the IEEE-488 bus. May happen on OPEN, CLOSE, CMD, INPUT#, GET#, PRINT#. If filename is not specified with OPEN, this error will occur.
6	?NOT INPUT FILE	An attempts was made to read a file originally OPENed for writing.
7	?NOT OUTPUT FILE	An attempts was made to write data to a file originally OPENed for reading. The keyboard cannot be written to.
8	?MISSING FILENAME	All LOADs and SAVEs from the IEEE port (eg. disk) require a filename.
9	?ILLEGAL DEVICE NUMBER	Occurs if you try to access a device in an illegal manner. For example, LOADing or SAVEing from/to the keyboard, screen, or RS-232.
10	?ARE YOU SURE	Confirmation prompt for BACKUP, SCRATCH, and HEADER. It is not an error message and occurs only in direct mode, not during BASIC program execution.
11	?BAD DISK	Media failure on HEADER command.
12	<return> READY. <return>	This Is Not An Error Message. This message lets you know that your system is ready to use.
13	<space> IN <space>	Not An Error Message. Used to indicate which line an error has occurred "in".
14	?BREAK	This occurs when the STOP key is pressed during BASIC execution. CONT can be used to restart the program.
15	?EXTRA IGNORED	Too many items of data or separators were entered in response to an INPUT statement.
16	?REDO FROM START	This diagnostic message occurs when a numeric variable is used with INPUT and non-numeric data is received. INPUT continues to function until acceptable data has been received.
17	Last Evaluated Number	This Is Not An Error Message. This is the last value that has been processed through the numerical output buffer. (eg. print 100/10 : print ERR$(17) ...will print 10 both times.
18	"MORE" <return>	This Is Not An Error Message. Prints "MORE" and carriage return.
19	Power On Message	This Is Not An Error Message. Prints the same screen message that is displayed immediately after power-up
20	?NEXT WITHOUT FOR	Either a NEXT is improperly nested or the variable in a NEXT statement corresponds to no previously executed FOR statement.
21	?SYNTAX	BASIC cannot recognize the statement you have typed. Caused by such things as missing parenthesis, illegal characters, incorrect punctuation, misspelled keyword.
22	?RETURN WITHOUT GOSUB	A RETURN statement was encountered with noprevious GOSUB.
23	?OUT OF DATA	An attempt was made to READ data from a DATA statement but no data exists or the program has already read them all.
24	?ILLEGAL QUANTITY	Occurs when a function is accessed with a parameter out of range caused by: 1. A matrix subscript out of range (0 < X < 32767) 2. LOG (negative or zero argument) 3. SQR (negative argument) 4. A↑B where A<0 and B not integer. 5. Call of USR before a machine language subroutine has been patched in. 6. Use of string functions MID$, LEFT$, RIGHT$, with length parameters out of range. 7. Index on...GOTO out of range. 8. Addressof PEEK, POKE, WAIT or SYS out of range. 9. Byte parameters of WAIT, POKE, TAB and SPC out of range.
25	?OVERFLOW	Numbers resulting from computations or input that are greater than 1.70141184E+38 or less than 2.93873587E-39.
26	?OUT OF MEMORY	BASIC text space, or Variables space, or Arrays memory space has been completely filled
27	?UNDEFINED STATEMENT	A GOTO, GOSUB, or THEN has been executed with a line number that does not exist.
28	?BAD SUBSCRIPT	An attempt was made to reference an array element which is outside the dimensions specified in the DIM statement.
29	?REDIM'D ARRAY	An attempt was made to define an array using a variable already used in an array.
30	?DIVISION BY ZERO	Illegal divide. Message is followed by the line number - list and check variables.
31	?ILLEGAL DIRECT	INPUT, INPUT#, GET, GET#, and DEF cannot be used in direct mode.
32	?TYPE MISMATCH	An arithmetic operation has been given non-numeric data, or a string operation has been numeric data.
33	?STRING TOO LONG	Maximum string length is 255 characters. This error will also occur if INPUT# receives more than 80 characters without a carriage return (ie. BASIC input buffer is 80 bytes long), or if a disk filename is longer than 16 characters.
34	?FILE DATA	Occurs when a numeric variable is used with INPUT# and non-numeric data is received.
35	?FORMULA TOO COMPLEX	BASIC has run out of temporary pointers to keep track of substrings in evaluating a string expression. Break the expression into two smaller parts to cure the problem.
37	?UNDEFINED FUNCTION	Reference was made to a user defined function which had never been defined with DEF.
38	?LOAD ERROR	Cassette tape only. To improve tape reliability, programs are recorded twice with SAVE. This error will occur if LOAD finds recording errors in corresponding positions of both recordings. If more than 31 errors are detected in the first pass, LOAD will not attempt to read the second.
39	?VERIFY ERROR	A VERIFY operation did not match the contents of file with the contents of memory. Re-SAVE your program on another disk or tape.
40	?OUT OF STACK	Too many open FOR...NEXT loops or too many GOSUB calls.
41	?UNABLE TO RESUME	Resume will not operate after a fatal error.
42	?UNABLE TO DISPOSE	All of the DISPOSE type items have been disposed of or none exist.
43	?OUT OF TEXT	A LOAD or DLOAD has attempted to bring in a file larger than 64K. This error will not occur when using the BLOAD command.

C64 Super Expander Commands, cont'd

Function	Example	Purpose
LOCATE	220 LOCATE X1, Y1	Set initial co-ordinates for plotting type commands to X1,Y1
MOVSPR	240 MOVSPR N, X, Y	Move Sprite N to X, Y
PAINT	PAINT C, X, Y, M	Fills the area surrounding X,Y in colour C using mode M. 0 = Bordered by same colour as C 1 = Bordered by any foreground colour
RBUMP	PRINT RBUMP (E)	Returns collision information for: 0 = Sprite to Sprite 1 = Sprite to Background
RCLR	PRINT RCLR (CS)	Returns Colour Source information for: 0 = Background colour number 1 = Foreground colour number 2 = Multi-Colour 1 colour number 3 = Multi-Colour 2 colour number 4 = Border colour number
RDOT	PRINT RDOT (M)	Returns information for the next pixel to be plotted using mode M. 0 = X co-ordinate 1 = Y co-ordinate 2 = Colour Source
RGR	PRINT RGR(0)	Returns GRAPHIC mode (0-3).
RJOY	PRINT RJOY(JS)	Returns direction (0-8) of Joystick 1 or 2. Fire Button adds 128 to direction value.
RPEN	PRINT RPEN(L)	Returns Location of Lightpen. 0 = X co-ordinate 1 = Y co-ordinate
RPOT	PRINT RPOT(P)	Returns Position (0-255) of Paddle P. 0 = Paddle 1 1 = Paddle 2 2 = Paddle 3 3 = Paddle 4 Fire Button adds 256 to position value
RSPCOL	PRINT RSPCOL(C)	Returns Spritecolour information. 0 = Multi-Colour 1 number 1 = Multi-Colour 2 number
RSPPOS	PRINT RSPPOS(SP,C)	Returns information for Sprite SP (0-7). C = 0 X co-ordinate C = 1 Y co-ordinate
RSPR	PRINT RSPR(SP,F)	Returns information for Sprite SP (0-7). F = 0 Sprite ON or OFF (1 or 0) F = 1 Foreground colour (0-15) F = 2 Display Priority (0 = above, 1 = below) F = 3 X Expand (1 = ON) F = 4 Y Expand (1 = ON) F = 5 Display mode (0 = Hi-Res, 1 = Multicolour)
SCALE	200 SCALE X	Set scale to: 0 = Standard co-ordinates based on GRAPHIC mode. 1 = Super Expander co-ordinate system.
SCNCLR	200 SCNCLR	Clears screen in any GRAPHIC mode
SPRCOL	200 SPRCOL M1, M2	Set sprite Multicolours 1 and 2 (0-15)
SPRDEF	SPRDEF	Enter Sprite Designer Function. Key detected are: 0-7 — Destination Sprite (prompted) A — Automatic Cursor movement toggle CRSR keys — Moves Cursor RETURN — Move to start of next line RETURN — Exit Sprite Designer (prompted) HOME — Move to Home position CLR — Erase grid 1-4 — Selects Colour Source CTRL 1-8 — Sprite Foreground Colour (0-7) Commodore 1-8 — Sprite Foreground Colour (8-15) STOP — Cancel changes Shift RETURN — Save Sprite X — X Expand Y — Y Expand M — Multi-Colour/Hi-Res toggle
SPRITE	200 SPRITE SP, EN, FG, PR, XE, YE, M	Set Sprite parameters. SP = Sprite number (0-7) EN = Enable (1 = ON) FG = Sprite Foreground colour (0-15) PR = Priority (0 = above, 1 = below) XE = X Expand (1 = ON) YE = Y Expand (1 = ON) M = Mode (0 = Hi-Res, 1 = Multi-Colour)
SPRSAV	200 SPRSAV SP, SP$	Save Sprite SP into SP$
SSHAPE	250 SSHAPE S$, X1, Y1, X2, Y2	Saves a shape into S$ from X2,Y2 to X1,Y1 (the diagonally opposite corner)
TEMPO	200 TEMPO T	Sets Tempo T = 0-255 (default 8)
TUNE	200 TUNE EV, AT, DC, SU, RL, WV, WT	Sounds note using: EV = Envelope number (0-9) AT = Attack rate (0-15) DC = Decay rate (0-15) SU = Sustain volume (0-15) RL = Release rate (0-15) WV = Waveform 0 = Triangle 1 = Sawtooth 2 = Pulse 3 = Noise 4 = Ring Modulation WT = Pulse Width (with WV = 2 only)

COMAL Commands

COMAL Flags & Reserved Variables

EOD	EOD	End Of Data flag
EOF	EOF(<filenum>)	End Of File flag
ESC	ESC TRAP ESC<type>	stop key pressed flag
FALSE	FALSE	predefined value = 0
STATUS$	STATUS$	status of disk channel
TRUE	TRUE	predefined value of 1

Note 1: Commodore BASIC, with the exception of a few commands, is a subset of COMAL. COMAL has all but ASC, CLR, DEF FN, GOSUB & RETURN, POS, REM, USR, VERIFY, WAIT, and BASIC 4.0 Disk Commands are sent via the COMAL PASS Command; other I/O commands (DLOAD, DCLOSE, RECORD*, etc) are much like BASIC 2.0 format.

Note 2: GOSUB (and ON...GOSUB) & RETURN are replaced by PROC Commands

Format: ( ) Numeric Brackets - numeric input required
< > Angle Brackets - denotes user supplied input
[ ] Square Brackets - indicates optional input

Thus: [(< >)] would specify the user supplied input must be of numeric nature, if the option is exercised.

Commands Common to COMAL and CBM BASIC With NO Differences

ABS	gives the absolute value
AND	logical AND
ATN	arctangent in radians
CHR$	gives that numbers character
COS	cosine in radians
DATA	provides data for a READ
END	halt program execution
EXP	natural log e to n
INT	gives nearest integer less than or equal
LEN	gives the length of string
LET	assign value to variable
LOG	natural logarithm of n
NEW	clears program from memory
NOT	logical NOT
OR	logical OR
PEEK	look at memory
POKE	change memory location
RESTORE	reuse DATA with READ
RUN	run program now in memory
SGN	-1 if neg, 0 if 0, 1 if pos
SIN	gives sine in radians
SQR	gives square root
STOP	halt program execution
SYS	transfer control to assembly language
TAB	print spaces up to specified column
TAN	gives tangent in radians
THEN	part of IF structure
TO	increment FOR variable start TO end

SPECIAL INFO

Line numbers allowed: 1-9999.
Identifiers up to 16 chars (unshifted alpha, digits, [,], ', <-, ↑)
Null input is accepted.

First time into graphics:	SETGRAPHIC 0
After that simply:	SETGRAPHIC

RUN/STOP RESTORE keys restore default colors.

To clean up the identifier name table:	LIST "PROGRAM.L"
(frees up memory,	NEW
removes unused identifiers)	ENTER "PROGRAM.L"
Save a program to disk:	SAVE "PROGRAM"
Load a program from disk:	LOAD "PROGRAM"
List a program to printer:	SELECT "LP:" LIST

COMAL 64 Colours List (COMAL 0.14/2.0)

Number	Colour	CHR$	Number	Colour	CHR$
0	BLACK	144	8	ORANGE	129
1	WHITE	5	9	BROWN	149
2	RED	28	10	LIGHT RED	150
3	CYAN	159	11	DARK GREY	151
4	PURPLE	156	12	MEDIUM GREY	152
5	GREEN	30	13	LIGHT GREEN	153
6	BLUE	31	14	LIGHT BLUE	154
7	YELLOW	158	15	LIGHT GREY	155

COMAL Commands NOT Found in CBM BASIC (* except BASIC 3.5)

***AUTO**	AUTO [<start line>][,<increment>]	automatic line numbering
BASIC	BASIC	back into BASIC mode
CASE	CASE <control expression> [OF]	multiple choice decisions
CHAIN	CHAIN <filename>	load & run program on disk
CLOSED	PROC <procname>[(params)] [CLOSED] FUNC <funcname>[(params)] [CLOSED]	all proc or func variables local
***DEL**	DEL <range>	deletes lines
DIV	<dividend> DIV <divisor>	division with integer answer
***DO**	DO <statements>	do the following statements
EDIT	EDIT [<range>]	lists lines without indentations
ELIF	ELIF <expression> [THEN]	short for ELSE IF condition
***ELSE**	ELSE	alternative statements in IF structure
ENDCASE	ENDCASE	end of CASE structure
ENDFOR	ENDFOR [<control variable>]	end of FOR structure
ENDFUNC	ENDFUNC [<function name>]	end of function
ENDIF	ENDIF	end of IF structure
ENDPROC	ENDPROC [<procedure name>]	end of procedure
ENDWHILE	ENDWHILE	end of WHILE structure
ENTER	ENTER <filename>	merge a program segment from disk
EXEC	[EXEC] <procname>[(<actual parameter list>)]	execute a procedure
FUNC	FUNC <name>[(<params>)] [EXTERNAL <filename>] FUNC <name>[(<params>)] [CLOSED]	start of a multiline function
IN	<string1> IN <string2>	locate position of string1 within string2
KEY$	KEY$	scans keyboard (not in PET COMAL 0.14)
LABEL	<label name>:	assigns a label name to the line
MOD	<dividend> MOD <divisor>	gives remainder of division (modulo)
NULL	NULL	does nothing (no op)
OF	CASE <expression> [OF] DIM <stringvar> OF <max char> DIM <stringarray>(array index) OF <max char>	part of DIM or CASE structure
OTHERWISE	OTHERWISE	default for CASE
PROC	PROC <name>[(<params>)] [EXTERNAL <filename>] PROC <name>[(<params>)] [CLOSED]	start of multiline procedure
RANDOM	OPEN FILE <filenum>,<filename>,RANDOM <recln>	random access disk file
RANDOMIZE	RANDOMIZE	generate new random numbers
REF	REF <var>	param var used in reference in proc
***RENUM**	RENUM [<targetstart>][,<increment>]	renumber program
REPEAT	REPEAT	start of REPEAT structure
***TRAP**	TRAP ESC<type>	disable stop key
***UNTIL**	UNTIL <expression>	end of REPEAT loop
***USING**	PRINT USING <format>: <var list> PRINT [FILE <filenum>:] USING <format>:<vars>	allows formatted output (not PET 0.14) including FILE output.
WHEN	WHEN <list of values>	choice in CASE structure
***WHILE**	WHILE <expression> [DO] [<statement>]	start of WHILE structure
WRITE	WRITE FILE <filenum>[,<recnum>]: <var list> OPEN [FILE] <filenum>,<filename>,WRITE	write to a file
ZONE	ZONE <tab interval> ZONE	tab increment

Commands Common to COMAL and CBM BASIC With SLIGHT Differences

//	//[<anything>]	allows comments in a program
APPEND	OPEN [FILE] <filenum>,<filename>,APPEND	start at end of seq file
CAT	CAT [<drive number>]	gives disk directory
CLOSE	CLOSE [FILE] [<filenum>]	closes files
CON	CON	continue program execution
DELETE	DELETE <filename>	deletes a file from disk
DIM	DIM <string var> OF <max char> DIM <str array>(<array index>) OF <max char> DIM <array name>(<array index>)	reserves/allocates string & array space
FILE	INPUT FILE <filenum>[,<recnum>]: <var list> PRINT FILE <filenum>[,<recnum>]: <val list> READ FILE <filenum>[,<recnum>]: <var list> WRITE FILE <filenum>[,<recnum>]: <var list> OPEN [FILE] <filenum>,<filename>[,<type>] CLOSE [FILE] [<filenum>]	specifies that a file is to be used
FOR	FOR <var> = <start> TO <end> [STEP <step>] [DO]	start of FOR loop structure
GOTO	GOTO <label name>	go to line with this name
IF	IF <condition> [THEN] IF <condition> THEN <statement>	start of conditional IF structure
INPUT	INPUT [<prompt>:] <var list> INPUT FILE <filenum>[,<recnum>]:<var list>	input from keyboard or file
LIST	LIST [<range>] [<filename>]	list program
LOAD	LOAD <filename>	load a program from disk
OPEN	OPEN [FILE] <filenum>,<filename>[,<type>]	open a file
ORD	ORD(<string expression>) (same as ASC(in BASIC)	returns integer representing the char
OUTPUT	SELECT [OUTPUT] <type>	select output location **Like CMD**
PASS	PASS <disk command>	passes a string to disk command channel
PRINT	PRINT [FILE <filenum>:] [<items>] PRINT [FILE <filenum>:] USING <format>:<vars> (RANDOM file use: [FILE <filnum>,<recnum>:])	prints items to screen/printer/file
READ	READ <var list> READ FILE <filenum>[,<rec num>]: <var list> OPEN [FILE] <filenum>,<filename>,READ	read data from DATA line or file
RND	RND(<num>) RND(<start num>:<end num>)	random number
SAVE	SAVE <filename>	record program on disk
SELECT	SELECT [OUTPUT] <type>	choose output location
SIZE	SIZE	reports on memory usage (free memory)
STEP	STEP <numeric expression>	increment FOR loop var by this amount
UNIT	OPEN FILE <*>,<nm>,UNIT <dev>[,<sec>][,<typ>]	specify unit (device)

SPRITES (COMAL 0.14/2.0)

DATACOLLISION	DATACOLLISION <sprite#>,<reset collisn flg?>	test for collision with data
DEFINE	DEFINE <sprite definition num>,<64 byte def$>	set up a sprite image for later use
HIDESPRITE	HIDESPRITE <sprite number>	turn off specified sprite
IDENTIFY	IDENTIFY <sprite number>,<definition number>	assign a sprite an image
PRIORITY	PRIORITY <sprite number>,<data priority?>	does data has priority over sprite
SPRITEBACK	SPRITEBACK <color1>,<color2>	set two multicolor sprite colors
SHOWSPRITE	SHOWSPRITE <sprite number>	turn on specified sprite
SPRITECOLLISION	SPRITECOLLISION <sprite#>,<reset collsn flg?>	test for sprite collision
SPRITECOLOR	SPRITECOLOR <sprite number>,<color number>	set color of sprite
SPRITEPOS	SPRITEPOS <sprite#>,<x coord>,<y coord>	position sprite at x,y location
SPRITESIZE	SPRITESIZE <sprite#>,<y expand?>,<x expand?>	set sprite size (expand or not)

HIGH RES and TURTLE Graphics (COMAL 0.14/2.0)

BACK	BACK <length>	move turtle backwards
BACKGROUND	BACKGROUND <color number>	set the screen background color
BORDER	BORDER <color number>	set the screen border color
CLEAR	CLEARSCREEN	clear the graphics screen (in frame)
DRAWTO	DRAWTO <x coordinate>,<y coordinate>	draws a line from current point
FILL	FILL <x coordinate>,<y coordinate>	fills in area with current color
FORWARD	FORWARD <length>	move turtle forward
FRAME	FRAME <x0>,<x1>,<y0>,<y1>	set up a screen window
FULLSCREEN	FULLSCREEN	fullscreen graphics (f5)
HIDETURTLE	HIDETURTLE	make the turtle invisible
HOME	HOME	put the turtle in its home position
LEFT	LEFT <degrees>	turn turtle left
MOVETO	MOVETO <x coordinate>,<y coordinate>	move to specified point without line
PENCOLOR	PENCOLOR <color number>	sets the current turtle pen color
PENDOWN	PENDOWN	put pen down, turtle draws line
PENUP	PENUP	pick up pen, turtle doesn't draw line
PLOT	PLOT <x coordinate>,<y coordinate>	plot a point in current color
PLOTTEXT	PLOTTEXT <x coord>,<y coord>,<text$>	print text on graphics screen
RIGHT	RIGHT <degrees>	turn turtle right
SETGRAPHIC	SETGRAPHIC [<type>]	turn on graphics screen
SETHEADING	SETHEADING <degree>	set turtle heading
SETTEXT	SETTEXT	turn on text screen (f1)
SETXY	SETXY <x coordinate>,<y coordinate>	set turtle x and y coordinates
SHOWTURTLE	SHOWTURTLE (note: sprite 7 is used for the turtle)	make turtle visible
SPLITSCREEN	SPLITSCREEN	2 text lines above graphics (f3)
TURTLESIZE	TURTLESIZE <size>	set turtle size (0 to 10)

TURTLE GRAPHICS CHART

Turtle Control:	CBM LOGO	CBM COMAL
Move forward length	FORWARD	FORWARD
Move backward length	BACK	BACK
Home turtle	HOME	HOME
Turn turtle left	LEFT	LEFT
Turn turtle right	RIGHT	RIGHT
Move to a point	SETXY	SETXY
Turn to specific heading	SETHEADING	SETHEADING
Make turtle visible	SHOWTURTLE	SHOWTURTLE
Make turtle invisible	HIDETURTLE	HIDETURTLE
Pen up off paper	PENUP	PENUP
Pen down on paper	PENDOWN	PENDOWN
Set pen color	PENCOLOR	PENCOLOR
Number of colors	16	16
Set size of turtle	–	TURTLESIZE
Plot a point	–	PLOT
Print text in graphics	?	PLOTTEXT
Screen And Colour Control:		
Set screen window	?	FRAME
Clear graphics screen	CLEARSCREEN	CLEAR
Set to graphics mode	DRAW	SETGRAPHIC
Set to text screen	NODRAW	SETTEXT
Set background color	BACKGROUND	BACKGROUND
Set border color	–	BORDER
Fill in an area	–	FILL
Full screen mode	FULLSCREEN	FULLSCREEN
Split screen mode	SPLITSCREEN	SPLITSCREEN
Function Key Actions:		
F1	TEXT SCREEN	TEXT SCREEN
F3	SPLITSCREEN	SPLITSCREEN
F5	FULLSCREEN	FULLSCREEN

COMAL 2.0 Library Descriptions

Library (page $80, $A59A-$BFF1):
A5C1 Sense routine

PACKAGE **english:**
A686 Init routine

PACKAGE **dansk:**
A68C Init routine

PACKAGE **system:**
CA2F Init routine
A80B PROC setprinter(str)
A96A PROC hardcopy(str)
A976 PROC setrecorddelay(int)
A97D PROC setpage(int)
A984 FUNC inkey
A9B6 FUNC free
A9C3 PROC keywords'in'upper'case(int)
A9C6 PROC names'in'upper'case(int)
A9C9 PROC quote'mode(int)
A9E1 FUNC currow
A9E9 FUNC curcol
A9F6 PROC textcolors(int,int,int)
AA34 PROC defkey(int,str)
AA7F PROC showkeys
AB21 PROC bell(int)
AB2D PROC serial(int)
A7FF PROC settime(str)
A805 FUNC gettime
A878 PROC getscreen(REF str)
A87B PROC setscreen(REF str)

Library (page $83, $800F-$C000):
8081 Sense routine

PACKAGE **graphics:**
8CDC Init routine
95CB PROC window(real,real,real,real)
8F15 PROC viewport(int,int,int,int)
8CA3 PROC drawto(real,real)
8ADA PROC draw(real,real)
8B06 PROC plot(real,real)
8C7C PROC moveto(real,real)
8AE8 PROC move(real,real)
A62A PROC circle(real,real,real)
A64F PROC arc(real,real,real,real,real)
A564 PROC arcl(real,real)
A55B PROC arcr(real,real)
9426 PROC textstyle(int,int,int,int)
9157 PROC plottext(real,real,str)
8D9B PROC pencolor(int)
8DBE PROC textcolor(int)
8FC3 FUNC getcolor(real,real)
A37B PROC fill(real,real)
A380 PROC paint(real,real)
9496 PROC background(int)
9483 PROC textbackground(int)
950B PROC border(int)
951E PROC textborder(int)
8E2A PROC graphicscreen(int)
90FC PROC textscreen
A25D PROC splitscreen
A258 PROC fullscreen
88FA PROC clearscreen
895E PROC clear
A23B PROC showturtle
A248 PROC hideturtle
A20F PROC turtlesize(real)
90A9 FUNC xcor
90D6 FUNC ycor
8CA3 PROC setxy(real,real)
904D PROC setheading(real)
9094 FUNC heading
903F PROC left(real)
903C PROC right(real)
901A PROC forward(real)
9017 PROC back(real)
9536 PROC penup
9542 PROC pendown
954E PROC home
9576 PROC wrap
9584 PROC nowrap
A8D7 FUNC inq(int)
AFD7 PROC savescreen(str)
B027 PROC loadscreen(str)
ADF4 PROC printscreen(str,int)

PACKAGE **turtle:**
8CE2 Init routine
9017 PROC bk(real)
9496 PROC bg(int)
88FA PROC cs
901A PROC fd(real)
A248 PROC ht
903F PROC lt(real)
8D9B PROC pc(int)
9542 PROC pd
9536 PROC pu
903C PROC rt(real)
904D PROC seth(real)
A23B PROC st
9483 PROC textbg(int)
95CB PROC window(real,real,real,real)
8F15 PROC viewport(int,int,int,int)
8CA3 PROC drawto(real,real)
8ADA PROC draw(real,real)
8B06 PROC plot(real,real)
8C7C PROC moveto(real,real)
8AE8 PROC move(real,real)
A62A PROC circle(real,real,real)
A64F PROC arc(real,real,real,real,real)
A564 PROC arcl(real,real)
A55B PROC arcr(real,real)
9426 PROC textstyle(int,int,int,int)
9157 PROC plottext(real,real,str)
8D9B PROC pencolor(int)
8DBE PROC textcolor(int)
8FC3 FUNC getcolor(real,real)
A37B PROC fill(real,real)
A380 PROC paint(real,real)
9496 PROC background(int)
9483 PROC textbackground(int)
950B PROC border(int)
951E PROC textborder(int)
8E2A PROC graphicscreen(int)
90FC PROC textscreen
A25D PROC splitscreen
A258 PROC fullscreen
88FA PROC clearscreen
895E PROC clear
A23B PROC showturtle
A248 PROC hideturtle
A20F PROC turtlesize(real)
90A9 FUNC xcor
90D6 FUNC ycor
8CA3 PROC setxy(real,real)
904D PROC setheading(real)
9094 FUNC heading
903F PROC left(real)
903C PROC right(real)
901A PROC forward(real)
9017 PROC back(real)
9536 PROC penup
9542 PROC pendown
954E PROC home
9576 PROC wrap
9584 PROC nowrap
A8D7 FUNC inq(int)
AFD7 PROC savescreen(str)
B027 PROC loadscreen(str)
ADF4 PROC printscreen(str,int)

PACKAGE **sprites:**
98B9 Init routine
9979 PROC define(int,str)
9B0D PROC identify(int,int)
99AC PROC spritecolor(int,int)
99BB PROC spritepos(int,int,int)
9A4A PROC spritesize(int,int,int)
9B46 PROC showsprite(int)
9B52 PROC hidesprite(int)
9A83 PROC spriteback(int,int)
9A93 FUNC spritecollision(int,int)
9A96 FUNC datacollision(int,int)
9ABF PROC priority(int,int)
AB54 PROC linkshape(int)
AB5A PROC loadshape(int,str)
AB6E PROC saveshape(int,str)
9B6F PROC movesprite(int,int,int,int,int)
9A11 PROC stopsprite(int)
9DFC PROC animate(int,str)
9D13 FUNC moving(int)
9D1F PROC startsprites
9CEB FUNC spritex(int)
9CFF FUNC spritey(int)
9D3F FUNC spriteinq(int,int)
9ECD PROC stampsprite(int)

PACKAGE **font:**
CA2F Init routine
ABD0 PROC linkfont
ABDF PROC loadfont(str)
AC49 PROC keepfont
ABF1 PROC savefont(str)
AC57 PROC getcharacter(int,int,REF str)
AC87 PROC putcharacter(int,int,str)

PACKAGE **sound:**
B287 Init routine
B2FE PROC note(int,str)
B3DE PROC pulse(int,int)
B3FA PROC gate(int,int)
B412 PROC soundtype(int,int)
B436 PROC ringmod(int,int)
B455 PROC sync(int,int)
B474 PROC adsr(int,int,int,int,int)
B4AD PROC filterfreq(int)
B4CD PROC resonance(int)
B4E6 PROC filter(int,int,int,int)
B508 PROC filtertype(int,int,int,int)
B52C PROC volume(int)
B543 FUNC env3
B549 FUNC osc3
B54F FUNC frequency(str)
B55B PROC setscore(int,REF int(),REF int(),REF int())
B59F PROC playscore(int,int,int)
B5CD PROC stopplay(int,int,int)
B5FC FUNC waitscore(int,int,int)
B2E3 PROC setfrequency(int,real)

PACKAGE **paddles:**
CA2F Init routine
B62C PROC paddle(int,REF real,REF real,REF real,REF real)

PACKAGE **joysticks:**
CA2F Init routine
B6B9 PROC joystick(int,REF real,REF real)

PACKAGE **lightpen:**
B77D Init routine
B7FA PROC offset(int,int)
B7D1 FUNC penon
B79B PROC readpen(REF real,REF real,REF real)
B820 PROC timeon(int)
B82A PROC delay(int)
B80D PROC accuracy(int,int)

Commodore 64 Cartridge COMAL 2.0 Memory Map

(Rev 2.01) ©1984 COMAL Users Group, U.S.A., Ltd

0000	0	D6510	6510 On-Chip Data-Direction Register
0001	1	R6510	6510 On-Chip 6-Bit I/O/Map-Register
0002 -0004	2-4	PRPROC	Chain of Local Names (prepass)
0005	5	INTEGR	Floating Point Work
0006	6	PAGE	Current Memory Map
0007 -0008	7-8	PAGEPT	Pointer used by Load/Store/Exec
0009	9	PAGEX	Overlay for Load/Store/Exec Routines
000A	10	PAGEY	Overlay used for control of Jump table
000B	11	P6510	Old C64-Overlay for control of Jump Table
000C	12	RESOL	Graphics Resolution
000D	13	GCOLH	Graphics Pencolor*16
		COMAL Variables	
000E -000F	14-15	LOCLPT	Chain of old Variable Descriptions
0010 -0011	16-17	FORPT	Stack Entry Chain
0012	18	SCTYPE	Type of Symbol from Scanner
0013	19	TANSGN	Tan Sign / Comparison Evaluation Flag
0014	20	CODE	Used to hold a generated code
0015	21	CPNT	Pointer to Code Buffer, CDBUF
0016 -0017	22-23	SPROG	Pointer to Start of Program
0018 -0019	24-25	SVARS	Pointer to Start of variable table
001A -001B	26-27	SSTACK	Pointer to Start of Stack
001C -001D	28-29	SMAX	Pointer to top of Memory
001E	30	EXINF	Inf for Result Expression from EXPR
001F	31	LNLEN	Length of Line to be Executed
0020	32	NPNT	Pointer to Name
0021	33	TPNT	Pointer to String
0022 -0023	34-35	INDEX1	Utility Pointer
0024 -0025	36-37	INDEX2	Utility pointer
0026	38	RESM1	Product Area for Multiplication
0027	39	RESM2	
0028	40	RESM3	
0029	41	RESM4	
002A	42	RESM5	
002B -002C	43-44	DATAPT	Current Data pointer
002D -002E	45-46	STOS	Pointer to Top of Stack
002F -0030	47-48	SFREE	Pointer to Free Area of VAR.RES
0031 -0032	49-50	PRGPNT	Pointer to Start of Line
0033	51	CODPNT	Pointer to Code During Execution
0034 -0035	52-53	SCLSD1	Old SFREE (closed)
0036 -0037	54-55	SCLSD2	Old STOS (closed)
0038	56	INF1	
0039	57	INF2	Used for Operand Checking
003A	58	INF3	
003B -003C	59-60	Q1	Short Span Work Areas
003D -003E	61-62	Q2	
003F -0040	63-64	Q3	
0041 -0042	65-66	Q4	
0043 -0044	67-68	Q5	
0045 -0046	69-70	COPY1	Work Space for Copy: From
0047 -0048	71-72	COPY2	Work Space for Copy: To
0049 -004A	73-74	COPY3	Work Space for Copy: Length
004B	75	BUS	0 = Bus Idle
004C	76	STINF	Information for Statement
			$01 = No Line Number
			$02 = Another Statement Follows
			$04 = After WHILE...DO
			$08 = After FOR...DO
			$10 = Statement Ended by Comment
			$20 = After IF...THEN
			$40 = After REPEAT...UNTIL
004D	77	EXCINF	Execution Information
			$02 = Escape is Trapped (STOP)
			$04 = Make call of COMAL Interrupt Handler
			$08 = Escape met (STOP)
			$10 = SRQ Enabled
			$20 = User Request Enabled
			$80 = Software SRQ Only
		Variables for Floating Point Packages	
004E -0053	78-83	TEMPF3	Misc. Floating Point Work Area
0054	84	ESCAPE	STOP Key Flag
0055	85		Not used
0056	86	OLDOV	Old Overflow (rounding)
0057 -005B	87-91	TEMPF1	Misc. Floating Point Work Area (5 bytes)
005C -0060	92-96	TEMPF2	Misc. Floating Point Work Area (5 bytes)
0061 -0066	97-102	AC1	Accum*1
			AC1+0 = Exponent
			AC1+1 = Mantissa 1
			AC1+2 = Mantissa 2
			AC1+3 = Mantissa 3
			AC1+4 = Mantissa 4
			AC1+5 = Sign
0067	103	DEGREE	Series Evaluation Constant pointer
0068	104	BITS	Accum*1: Hi-order (overflow)
0069 -006E	105-110	AC2	Accum*2
			AC2+0 = Exponent
			AC2+1 = Mantissa 1
			AC2+2 = Mantissa 2
			AC2+3 = Mantissa 3
			AC2+4 = Mantissa 4
			AC2+5 = Sign
006F	111	ARISGN	Sign Comparison, Acc*1 vs Acc*2
0070	112	FACOV	Accum*1: Lo-order (rounding)
0071 -0072	113-114	POLYPT	Pointer to Polynomial
		More COMAL Variables	
0073	115	ASAVE	Save for .A (call/goto)
0074	116	XSAVE	Save for .X (call/goto)
0075	117	PSAVE	Save for .p (call/goto)
0076	118	INDPNT	Pointer to last code where an address was loaded
0077	119	SCFLAG	Flags in Scanner
0078 -0079	120-121	LNNO	Line Number
007A -007B	122-123	MOVEAD	Address for Move
007C	124	TXTLO	Address of Text for PRTXT
007D	125	TXTHI	
007E -007F	126-127	XX	Current X (graphics)
0080 -0081	128-129	YY	Current Y (graphics)
0082 -0083	130-131	GRWK1	
0084 -0085	132-133	GRWK2	

0086 -0087	134-135	GRWK3	
0088	136	EXCFLG	Flags: $01 = New Name has been inserted
			$02 = New Line has been inserted
0089	137	CHARPT	Pointer to INBUF
008A	138	CHAR	Char from INCHAR
008B -008F	139-143	RNDX	Random Number Seed
		Variables for I/O	
0090	144	STATUS	I/O Operation Status
0091	145	STKEY	STOP Key Flag
0092	146	SVXT	Temporary
0093	147	VERCK	Load or Verify Flag
0094	148	C3P0	IEEE Buffered Char Flag
0095	149	BSOUR	Char Buffer for IEEE
0096	150	SYNO	Cassette Sync *
0097	151	XSAV	Temp for BASIN
0098	152	LDTND	How many Files Open
0099	153	DFLTN	Default Input Device *
009A	154	DFLTO	Default Output Device *
009B	155	PRTY	Cassette Parity
009C	156	DPSW	Cassette Dipole Switch
009D	157	MSGFLG	OS Message Flag
009E	158	PTR1	Cassette Error Pass 1
009F	159	PTR2	Cassette Error Pass 2
00A0 -00A2	160-162	TIME	24 Hour Clock in 1/60 sec.
00A3	163	PCNTR	Serial Bus usage/Cassette stuff
00A4	164	FIRT	
00A5	165	CNTDN	Cassette sync countdown/temp used by serial routine
00A6	166	BUFPT	Tape Buffer Pointer
00A7	167	INBIT	RS232 Receiver Input bit storage/Cassette short count
00A8	168	BITCI	RS232 Receiver bit count in/Cassette read error
00A9	169	RINONE	RS232 Receiver Flag for start bit check/Cassette reading zeroes
00AA	170	RIDATA	RS232 Receiver byte buffer/Cassette read mode
00AB	171	RIPRTY	RS232 Receiver parity storage
00AC	172	SAL	Pointer: Tape Buffer/Screen Scrolling/Cassette short count
00AD	173	SAH	
00AE	174	EAL	
00AF	175	EAH	
00B0	176	CMP0	Tape Timing Constant
00B1	177	TEMP	Tape Timing Constant
00B2 -00B3	178-179	TAPE1	Start of Tape Buffer
00B4	180	BITTS	RS232 Transmit bit count/Cassette stuff
00B5	181	NXTBIT	RS232 Transmit next bit to be sent
00B6	182	RODATA	RS232 Transmit byte buffer/EOT received from tape
00B7	183	FNLEN	Length of Current File Name
00B8	184	LA	Current File Logical Address
00B9	185	SA	Current File Secondary Address
00BA	186	FA	Current File Primary Address
00BB -00BC	187-188	FILADR	Current File Name Address
00BD	189	ROPRTY	RS232 Transmit Parity Buffer
00BE	190	FSBLK	Cassette Read Block Count
00BF	191	MYCH	Serial word Buffer
00C0	192	CAS1	Cassette Manual/Controlled Switch
00C1 -00C2	193-194	STAL	Tape Start Address Low,High
00C3 -00C4	195-196	MEMUSS	Tape Load temps
		Variables for Screen Editor	
00C5	197	LSTX	Key Scan Index
00C6	198	NDX	Key Buffer Pointer
00C7	199	RVS	Reverse Field ON Flag
00C8	200	INDX	Byte Pointer to End of Line for Input
00C9	201	LSXP	Start of Screen Input (row)
00CA	202	LSTP	Start of Screen Input (column)
00CB	203	SFDX	Shift Mode on Print
00CC	204	BLNSW	Cursor Blink Enable
00CD	205	BLNCT	Counter to flip Cursor
00CE	206	GDBLN	Old Char before blink
00CF	207	BLNON	ON/OFF Blink Flag
00D0	208	CRSW	Input/Get Flag
00D1 -00D2	209-210	PNT	Pointer to Start of Line where Cursor is flashing
00D3	211	PNTR	Column Position where Cursor is flashing
00D4	212	QTSW	Flag for Quote Mode
00D5	213	LNMX	Current Screen Line Length (39/79)
00D6	214	TBLX	Line Number where Cursor is flashing
00D7	215	DATA	temp Data Area
00D8	216	INSRT	Number of Insert Keys pushed to go
00D9 -00F2	217-242	WRPTB	Line flags + endspace
00F3 -00F4	243-244	USER	Screen Editor Color Pointer
00F5 -00F6	245-246	KEYTAB	Keyboard Decode table
00F7 -00F8	247-248	RIBUF	RS232 Input Buffer Address
00F9 -00FA	249-250	ROBUF	RS232 Output Buffer Address
00FB -00FF	251-255	FREKZP	Free Kernal Zero Page Space
0100 -01FF	256-511	STACK	System Stack
0100 -010E	256-270	FBUFFR	FPASC Work Area (15 bytes)
0100	256	BAD	Tape Input Error Log
0200	512	ERTLEN	Length of ERTEXT, max. length of ERTEXT = 79
0201 -024F	513-591	ERTEXT	Buffer to hold Error Message, max. len. 79
		Storage for CON Command	
0250 -0251	592-593	CONPNT	Old PRGPNT
0252	594	CONFLG	Old EXCINF
0253	595	CONCOD	Old CODPNT
0254 -0255	596-597	CONFOR	Old FORPT
0256	598	FPWORK	
0257	599	EXTROM	External ROM Flag (0 = no, '1' = yes)
0258	600	IEEEIN	IEEE Installed (0 = no, '1' = yes)
0259 -0262	601-610	LAT	Table of Logical Addresses
0263 -026C	611-620	FAT	Table of File Addresses
026D -0276	621-630	SAT	Table of Secondary Addresses
0277 -0280	631-640	KEYBUF	Keyboard Buffer Queue (fifo)
0281 -0282	641-642	MEMSTR	Start of Memory
0283 -0284	643-644	MEMSIZ	Top of Memory
0285	645	TIMOUT	IEEE Time Out Defeat
		Screen Editor Storage	
0286	646	COLOR	Active Color nybble
0287	647	GDCOL	Original Color Under Cursor
0288	648	HIBASE	Base Location of Screen
0289	649	KBFLIM	Size of Keyboard Buffer
028A	650	RPTFLG	Key Repeat Flag
028B	651	RPTCNT	Repeat Speed Counter

028C	652	DELAY	Repeat Delay Counter
028D	653	SHFLAG	Keyboard Shift Key/Ctrl Key/Commodore Key
028E	654	LSTSHF	Last Keyboard Shift Pattern
028F -0290	655-656	KEYLOG	Vector: Keyboard table Setup
0291	657	SHMODE	0 = PET Mode, 1 = Cattacanna
0292	658	AUTODN	Auto Scroll Down, 0 = ON
		RS232 Storage	
0293	659	M51CTR	6551 Control Register Image
0294	660	M51CDR	6551 Command Register Image
0295 -0296	661-662	M51AJB	Non-Standard BPS (time/2-100) USA
0297	663	RSSTAT	6551 Status Register
0298	664	BITNUM	Number of bits left to send
0299 -029A	665-666	BAUDOF	Baud Rate: full bit time (microsec)
029B	667	RIDBE	Index to End of Input Buffer
029C	668	RIDBS	Start of Input Buffer (page)
029D	669	RODBS	Start of Output Buffer (page)
029E	670	RODBE	Index to End of Output Buffer
029F -02A0	671-672	IRQTMP	Holds IRQ-Vector during Tape I/O
		Temporary Space for C64 Variables	
02A1	673	ENABL	RS232 Enables
02A2	674	CASTON	TOD Sense during Cassette I/O
02A3	675	KIKA26	Temp Storage for Cassette read
02A4	676	STUPID	Temp D1IRQ Indicator for Cassette read
02A5	677	LINTMP	Temp for Line Index
02A6	678	PALNTS	Flag: 0 = NTSC, 1 = PAL
02A7 -02DD	679-733	FILNAM	used for Storage of File Name/Disk Commands
02DE	734	RANGNO	Line * Range Pointer
02DF	735	RANGPT	Line * Range Pointer
02E0 -02FF	736-767	RANGES	Line * Ranges, max. 32
0300 -0301	768-769	IERROR	Vector: Print Error Message
0302 -0306	770-774		Not used
0307 -030B	775-779	NUM2	Floating Point Work Area (PRINT USING)
030C -0313	780-787	SAREG	Unused
		Kernal Vectors	
0314 -0315	788-789	CINV	IRQ RAM Vector
0316 -0317	790-791	CBINV	BRK Instr RAM Vector
0318 -0319	792-793	NMIVCT	NMI RAM Vector
031A -031B	794-795	IOPEN	OPEN Routine Vector
031C -031D	796-797	ICLOSE	CLOSE Routine Vector
031E -031F	798-799	ICHKIN	CHKIN Routine Vector
0320 -0321	800-801	ICKOUT	CKOUT Routine Vector
0322 -0323	802-803	ICLRCH	CLRCHN Routine Vector
0324 -0325	804-805	IBASIN	CHRIN Routine Vector
0326 -0327	806-807	IBSOUT	CHROUT Routine Vector
0328 -0329	808-809	ISTOP	STOP Routine Vector
032A -032B	810-811	IGETIN	GETIN Routine Vector
032C -032D	812-813	ICLALL	CLALL Routine Vector
032E -032F	814-815	USRCMD	For Machine Language Monitor
0330 -0331	816-817	ILOAD	LOAD Routine Vector
0332 -0333	818-819	ISAVE	SAVE Routine Vector
0334 -033B	820-827		Unused
033C -03FB	828-1019	TBUFFR	Tape I/O Buffer
03FC -03FF	1020-1023		Unused
0400 -07E7	1024-2023	SCREEN	Screen Memory Area (1000 bytes)
07E8 -07F7	2024-2039		Screen Memory Excess
07F8 -07FF	2040-2047	SPRPNT	Sprite Data Pointers
		COMAL Program Follows Here	
0800 -0803	2048-2051	MBEGIN	Start of Memory
0804	2052	MBEGN1	Start of Name table
0805	2053	MBEGN2	Start of Stacks
C000 -C87A	49152-51322		Additional COMAL Storage
C87B -CA4A	51323-51786		COMAL Subroutines
D000 -D027	53248-53287		6566 VIC II Video Interface
D400 -D41C	54272-54300		6581 SID Sound Interface
D500 -D7FF	54528-55295	SIDIMG	SID Images
D800 -DBFF	55296-56319	COLRAM	Color RAM (nybbles)
DC00 -DC0F	56320-56335		6526 Complex Interface Adapter *1
DD00 -DD0F	56576-56591		6526 Complex Interface Adapter *2
DE00 -DEFF	56832-67087	OVERLAY	Overlay Control Port
FF81 -FFF9	65409-65529		Kernal Jump Table
C000 -C0FF	49152-49407	RSIBUF	RS232 Input Buffer
C100 -C1FF	49408-49663	RSOBUF	RS232 Output Buffer
C200 -C5E7	49664-50663	STDPCK	Variables for standard packages
C5E8 -C660	50664-50784	INBUF	Input Buffer (121 bytes)
C661 -C75F	50785-51039	CDBUF	Code Buffer
C760 -C7AF	51040-51119	TXT	String Constant Buffer
C7B0	51120	FLEVEL	FOR/TRAP Nesting Level during prepass
C7B1 -C7B2	51121-51122	Q6	Temporary
C7B3 -C7B4	51123-51124	Q7	Temporary
C7B5 -C7B6	51125-51126	Q8	Temporary
C7B7 -C7B8	51127-51128	Q9	Temporary
C7B9 -C7BB	51129-51131		Unused
C7BC	51132	SPSAV	Save of .S during execution
C7BD -C7BE	51133-51134	SCINF	Name Pointer
C7BF -C7C0	51135-51136	AUTO1	Current Line Number for AUTO
C7C1 -C7C2	51137-51138	AUTOST	Step for AUTO
C7C3 -C7C4	51139-51140	DSTART	Start of Data Queue
C7C5	51141	TABSET	value of Last ZONE Statement
C7C6	51142	ALTPOS	Position in Select Output File
C7C7 -C7C8	51143-51144	INTRNO	Procedure given by Interrupt Statement
C7C9	51145	ERRPNT	Character Position of Error
C7CA -C7CB	51146-51147	NORINT	Normal Interrupt vector
C7CC	51148	SAFE	Safe Status
C7CD	51149	MAINRV	Main Revision
C7CE	51150	SUBRV	Sub Revision
C7CF	51151	TESTRV	Test Version
C7D0 -C7D1	51152-51153	MSGLIN	Address of Message Line
C7D2	51154	UPPER2	Copy of Borge
C7D3	51155	EXTPRC	Flag for Loading of External PROC/FUNC
C7D4 -C7D5	51156-51157	EXTCNT	Nesting Level of External PROC/FUNC
C7D6	51158	SSIZE	40 Col./80 Col
C7D7	51159	LUNIT	Last Disc (status)
C7D8	51160	BORGE	Special Flags for Listing
C7D9	51161	OPENFL	Flag used by COPEN
C7DA	51162	DFUNLN	Length of Default Unit Text
C7DB -C7DC	51163-51164	DFUNIT	Default Unit (power up value- byte 0)
C7DD	51165	DEFOUT	Select Output Flag
C7DE -C7DF	51166-51167	TRAPVC	Page B. Error Handler
C7E0 -C7E1	51168-51169	EXTNVC	Page B. External Load
C7E2 -C7E3	51170-51171	USRQVC	Page B. Interrupt Facility
C7E4 -C7E6	51172-51174	IERTXT	Error Message Data

C7E7 -C7E8	51175-51176	IGETLN	Page A: Input Command Line
C7E9 -C7EA	51177-51178	ISAVEC	Page C: Save Additional Info
C7EB -C7EC	51179-51180	ILOADC	Page C: Load Additional Info
C7ED -C7EE	51181-51182	IFNKEY	Page A: Handle Function Keys
C7EF	51183	LIBPT	Pointer: to Place for Next Library Descrip.
C7F0 -C7F9	51184-51193	LIBLO	Library Descriptions, max. 10
C7FA -C803	51194-51203	LIBHI	
C804 -C80D	51204-51213	LIBPAG	
C80E -C817	51214-51223	MODET	Open Mode for Files
C818 -C821	51224-51233	COUNTT	Table of Byte Count for Files
C822 -C82B	51234-51243	STT	Status for Opened Files
C82C -C835	51244-51253	RECOTL	Table of Record Position for Files
C836 -C83F	51254-51263	RECOTH	
C840	51264	PPAGE	Overlay to PEEK/POKE/SYS
C841	51265	NOREST	<>0- Disable STOP/Restore
C842	51266	LOADIN	<>0- Loading COMAL Program
C843	51267	UNITFL	0-simp.dev, 1-Drive, 2-Cassette
C844	51268	MODE	File Mode
C845	51269	CSTAT	Status of COMAL Program
			1 = Input analysis from screen
			2 = Input analysis from file
			3 = Prepassing
			4 = Executing a command
			5 = Executing program
C846	51270	LSTFLG	Bit Vector for RCREAT
C847	51271	LPMODE	Default Printer Open Mode
C848	51272	LPSA	Default Printer Secondary Address
C849	51273	LPFA	Default Printer Unit
C84A	51274	RECDEL	Record Positioning Delay
C84B -C84C	51275-51276	ENDADR	Top of RAM
C84D	51277	HEADLN	Power On Message Flag
C84E -C84F	51278-51279	KWTAB	Keyword Table (Page A)
C850	51280	DFBORD	Default Border Color
C851	51281	DFBACK	Default Background Color
C852	51282	DFFORG	Default Foreground Color
C853	51283	ACBORD	Actual Text Border
C854	51284	ACBACK	Actual Text Background
C855 -C864	51285-51300	KEYLEN	Lengths of Function Key def's
C865	51301	KLEN	* of Chars left of Define
C866 -C867	51302-51303	KPNT	Pointer to Key Def
C868	51304	DEFINP	Select Input Flag
C869	51305	HZ50	0 = 60 Hz, 1 = 50 Hz TOD
C86A -C87A	51306-51322		Reserved for future use
		Subroutines to use in COMAL Assembler Routines	
C87B -C87D	51323-51325	COLD	Cold Start of COMAL
C87E -C880	51326-51328	WARM	Warm Start of COMAL
C881 -C883	51329-51331	CALL	JSR to another page.
C884 -C886	51332-51334	GOTO	JMP to another page.
C887 -C889	51335-51337	LOAD	Load from Page X
C88A -C88C	51338-51340	STORE	Store to Page X
C88D -C88F	51341-51343	EXEC	JSR to Page X
C890 -C892	51344-51346	LDAC1	Load Ac1
C893 -C895	51347-51349	LDAC2	Load Ac2
C896 -C898	51350-51352	FNDPAR	Find Parameter (asm calls)
C899 -C8A1	51353-51361	COPY	Copy Area towards lower addresses
C8A2 -C8AA	51362-51370	COPYDN	Copy Area towards higher addresses
C8AB -C8AD	51371-51373	FPADD	Load Ac2 and add Ac2 to Ac1
C8AE -C8B6	51374-51382	FPADD2	Add Ac2 to Ac1
C8B7 -C8BF	51383-51391	FPAHF	Add 0.5 to Ac1
C8C0 -C8C2	51392-51394	FPSUB	Load Ac2 and sub Ac2 from Ac1
C8C3 -C8CB	51395-51403	FPSUB2	Sub Ac2 from Ac1
C8CC -C8CE	51404-51406	FPMUL	Load Ac2 and mult Ac2 by Ac1
C8CF -C8D7	51407-51415	FPMUL2	Mult Ac2 by Ac1
C8D8 -C8DA	51416-51418	FPDIV	Load Ac2 and div Ac2 by Ac1
C8DB -C8E3	51419-51427	FPDIV2	Div Ac2 by Ac1
C8E4 -C8EC	51428-51436	MUL10	Multiply Ac1 by 10.0
C8ED -C8F5	51437-51445	DIV10	Divide Ac1 by 10.0
C8F6 -C8F8	51446-51448	STAC1	Store AC1
C8F9 -C901	51449-51457	C1T2	Copy Ac1 to Ac2
C902 -C90A	51458-51466	C2T1	Copy Ac2 to Ac1
C90B -C913	51467-51475	FPNEG	Negate Ac1
C914 -C91C	51476-51484	FPSGN	Sign of Ac1
C91D -C925	51485-51493	FPSIN	Sine of Ac1
C926 -C92E	51494-51502	FPCOS	Cosine of Ac1
C92F -C937	51503-51511	FPSQR	Square root of Ac1
C938 -C940	51512-51520	FPTAN	Tangent of Ac1
C941 -C949	51521-51529	FPPOW	Raise Ac2 to the power of Ac1
C94A -C952	51530-51538	FPATN	Arctangent of Ac1
C953 -C95B	51539-51547	FPEXP	Raise Ac1 to the power of e
C95C -C964	51548-51556	FPLOG	Logarithm Base e of Ac1
C965 -C96D	51557-51565	FPRND	Compute pseudo-random number (0 to 1)
C96E -C976	51566-51574	FPCOM	Compare Number to Ac1
C977 -C97F	51575-51583	TRUNC	Convert Ac1 to Integer (-32768-32767)
C980 -C988	51584-51592	FPINTG	Convert Ac1 to Integer (-2↑24-2↑24-1)
C989 -C991	51593-51601	FPINTA	Convert Ac1 to Integer (0-65535)
C992 -C99A	51602-51610	INTFP	Convert Integer to Floating Point in Ac1
C99B -C9A3	51611-51619	FPASC	Convert Ac1 to ASCII equiv. (STR$)
C9A4 -C9A9	51620-51625	VAL	Convert Decimal string to Binary in Ac1
C9AA -C9B2	51626-51634	POPA1	Pop Ac1
C9B3 -C9BB	51635-51643	POPA2	Pop Ac2
C9BC -C9C4	51644-51652	PUSHA1	Push Ac1
C9C5 -C9CD	51653-51661	PUSHRI	Push Real Number
C9CE -C9D6	51662-51670	PSHINT	Float & Push Integer (-32768 32767)
C9D7 -C9DF	51671-51679	INTFPA	Float & Push Integer (0 65535)
C9E0 -C9E8	51680-51688	EXCGST	Allocate Local Storage
C9E9 -C9F1	51689-51697	EXCREM	Reclaim Local Storage
C9F2 -C9FA	51698-51706	RESTOP	Allocate Global Storage
C9FB -CA00	51707-51712	RUNERR	Go to COMAL Error Handler
CA01 -CA03	51713-51715	CRDT	Read Character
CA04 -CA05	51716-51717	SPACE	Write Space
CA06 -CA08	51718-51720	CWRT	Write Character
CA09 -CA0B	51721-51723	CCHKIN	Select Input File
CA0C -CA0E	51724-51726	CCKOUT	Select Output File
CA0F -CA11	51727-51729	CCLRCH	Clear Channel
CA12 -CA18	51730-51736	CFNAME	Parse & Copy file name
CA19 -CA1B	51737-51739	COPEN	Open File
CA1C -CA1E	51740-51742	CCLOSE	Close File
CA1F -CA21	51743-51745	CRLF	Output CR and LF
CA22 -CA28	51746-51752	GETLIN	Input Keyboard Line
CA29 -CA2E	51753-51758	RESET	Reset Program Pointers
CA2F	51759	DUMMY	Empty Subroutine (RTS)
CA30 -CA35	51760-51765	COMAL	Go to COMAL Editor

CA36 -CA3C	51766-51772	EXCUTE	Execute Code in CDBUF
CA3D -CA43	51773-51779	JLOAD	Load COMAL Program
CA44 -CA4A	51780-51786	ARRLEN	Compute * of Array Elements

6566 VIC II Video Interface Controller

D000	53248	SPRPOS	Sprites 0-7 X & Y Position
D010	53264	SPRXPS	Sprites 0-7 X Position (msb of X-coord.)
D011	53265	VCTRL1	VIC Control Register
D012	53266	RWRAST	Read/Write Raster Value for compare IRQ
D013	53267	PENX	Light-Pen Latch X Position
D014	53268	PENY	Light-Pen Latch Y Position
D015	53269	SPRDSP	Sprite Display Enable
D016	53270	VCTRL2	VIC Control Register
D017	53271	SPRYEX	Sprites 0-7 expand 2* vertical (Y)
D018	53272	VCTRL3	VIC Memory Control Register
D019	53273	IRQOCC	VIC Interrupt Flag Register
D01A	53274	IRQMSK	IRQ Mask Register
D01B	53275	SPRBDP	Sprite to Background display priority
D01C	53276	SPRMCM	Sprites 0-7 Multi-Color mode select
D01D	53277	SPRXEX	Sprites 0-7 expand 2* horizontal (x)
D01E	53278	SPRSPR	Sprite to Sprite collision detect
D01F	53279	SPRBCK	Sprite to Background collision detect
D020	53280	BORCOL	Border Color
D021	53281	BCKCOL	Background Color 0-3
D025	53285	SPRMCL	Sprite Multi-Color Register 0-1
D027	53287	SPRCOL	Sprite 0-7 Color

6581 SID Sound Interface Device

D400	54272	V1FREQ	**Voice 1** Frequency Control
D402	54274	V1PWW	Pulse Waveform Width
D404	54276	V1CTRL	Control Register
D405	54277	V1ENVL	Envelope Generator (adsr)
D407	54279	V2FREQ	**Voice 2** Frequency Control
D409	54281	V2PWW	Pulse Waveform Width
D40B	54283	V2CTRL	Control Register
D40C	54284	V2ENVL	Envelope Generator (adsr)
D40E	54286	V3FREQ	**Voice 3** Frequency Control
D410	54288	V3PWW	Pulse Waveform Width
D412	54290	V3CTRL	Control Register
D413	54291	V3ENVL	Envelope Generator (adsr)
D415	54293	FCUTOF	Filter Cutoff Frequency
D417	54295	FRESON	Filter Resonance/Voice Input Control
D418	54296	FMOVOL	Select Filter Mode and Volume
D419	54297	PADDL1	A/D-Converter Game Paddle 1
D41A	54298	PADDL2	A/D-Converter Game Paddle 2
D41B	54299	OSC	Oscillator 3 Random Number Generator
D41C	54300	ENV	Envelope Generator 3 Output
D500		SIDIMG	SID Images
D800		COLRAM	Color RAM (nybbles)

	6526 CIA *1			**6526 CIA *2**
DC00	COLM	Keyboard Matrix PRA	DC00	D2PRA
DC01	ROWS	Keyboard Matrix PRB	DC01	D2PRB
DC02	D1DDRA		DC02	D2DDRA
DC03	D1DDRB		DC03	D2DDRB
DC04	D1T1L		DC04	D2T1L
DC05	D1T1H		DC05	D2T1H
DC06	D1T2L		DC06	D2T2L
DC07	D1T2H		DC07	D2T2H
DC08	D1TOD1		DC08	D2TOD1
DC09	D1TODS		DC09	D2TODS
DC0A	D1TODM		DC0A	D2TODM
DC0B	D1TODH		DC0B	D2TODH
DC0C	D1SDR		DC0C	D2SDR
DC0D	D1ICR		DC0D	D2ICR
DC0E	D1CRA		DC0E	D2CRA
DC0F	D1CRB		DC0F	D2CRB

DE00	OVERLAY	Overlay Control Port

Kernal Jump Table

FF81	CINT	Init Screen Editor
FF84	IOINIT	Init I/O
FF87	RAMTAS	Init RAM
FF8A	RESTOR	Restore Default I/O Vectors
FF8D	VECTOR	Read/Set Vectored I/O
FF90	SETMSG	Control Kernal Messages
FF93	SECOND	Send SA after Listen
FF96	TKSA	Send SA after Talk
FF99	MEMTOP	Read/Set Top of Memory
FF9C	MEMBOT	Read/Set Bottom of Memory
FF9F	SCNKEY	Scan Keyboard
FFA2	SETTMO	Set Timeout Defeat
FFA5	ACPTR	Input byte from Serial Port
FFA8	CIOUT	Output byte to Serial Port
FFAB	UNTLK	Command device to Untalk
FFAE	UNLSN	Command device to Unlisten
FFB1	LISTN	Command device to Listen
FFB4	TALK	Command device to Talk
FFB7	RDSTAT	Read I/O Status Word
FFBA	SETLFS	Set Logical, First, and Second Addr
FFBD	SETNAM	Set File Name
FFC0	OPEN	Open Logical File
FFC3	CLOSE	Close Logical File
FFC6	CHKIN	Open Channel for Input
FFC9	CHKOUT	Open Channel for Output
FFCC	CLRCHN	Close I/O channels
FFCF	CHRIN	Input Char from channel
FFD2	CHROUT	Output Char to channel
FFD5	LOADF	Load RAM from a Device
FFD8	SAVEF	Save RAM to Device
FFDB	SETTIM	Set Real Time Clock
FFDE	RDTIM	Read Real Time Clock
FFE1	STOP	Scan STOP Key
FFE4	GET	Get Char from Keyboard Buffer
FFE7	CLALL	Close all channels and files
FFEA	UDTIM	Increment Real Time Clock
FFED	SCREN	Return Row,Column of Cursor
FFF0	PLOT	Read/Set Row Column for Cursor
FFF3	IOBASE	Returns Base Address of I/O Devices
FFFA	CNMI	Non-Maskable Interrupt Vector
FFFC	CRESET	System Reset Vector
FFFE	CIRQ	Interrupt Request Vector

Commodore 64 COMAL 0.14 Memory Map

0000	0	6510 On-Chip Data Direction Register
0001	1	6510 On-Chip 5-bit Input/Output Register
002B	43	Temporary Storage of Error Number about to be generated
0038 -0039	56–57	Start of Program (start value 35153)
003A -003B	58–59	Start of Variables (start value 35153)
003C -003D	60–61	Start of Name Table (start value 35153)
003E -003F	62–63	End of Name Table (start value 35154)
0040 -0041	64–65	Start of Variables (start value 35161)
0042 -0043	66–67	Bottom of DIM Variables (start value 45056)
		(reset by NEW/RUN/chain) (reset takes value from 2066–2067)
0044 -0045	68–69	Highest Location used by COMAL (start value 45056)
		(reset by NEW/chain) (reset takes value from 2066–2067)
0061	97	Floating Point Accumulator*1. Exponent
0062 -0065	98–101	Floating Point Accumulator*1: Mantissa
0066	102	Floating Accumulator*1: Sign
0067	103	**Pointer:** Series Evaluation Constant
0068	104	Floating Point Accumulator*1: Overflow Digit
0069	105	Floating Point Accumulator*2: Exponent
006A -006D	106–109	Floating Accumulator*2: Mantissa
006E	110	Floating Point Accumulator*2: Sign
006F	111	Sign Comparison Result: Accum.*1 versus *2
0070	112	Floating Accumulator*1: Low-Order (Rounding)
0071 -0072	113–114	Pointer to the Cassette Buffer
0090	144	Kernal I/O Status Word
0091	145	Reverse Field (0 = off 1 = on)
0092	146	Timing Constant for Tape
0093	147	**Flag:** 0 = Load, 1 = Verify
0094	148	**Flag:** Serial Bus-Output Char. Buffered
0095	149	Buffered Char. for Serial Bus
0096	150	Cassette Sync Number
0097	151	Temp Data Area
0098	152	0 = screen 1 = printer // Output Location – see also 4348
0099	153	Default Input Device (0)
009A	154	Default Ouput Device (3)
009B	155	Tape Character Parity
009C	156	**Flag:** Tape Byte-Recieved
009E	158	Tape Pass 1 Error Log
009F	159	Tape Pass 2 Error Log
00A0 -00A2	160–162	Real Time Jiffy Clock
00A5	165	Cassette Sync Countdown
00A6	166	**Pointer:** Tape I/O Buffer
00A7	167	RS-232 Input Bits / Cassette Temp
00A8	168	RS-232 Bit Count / Cassette Temp
00A9	169	RS-232 **Flag:** Check for Start Bit
00AA	170	RS-232 Input Byte Buffer / Cassette Temp
00AB	171	RS-232 Input Parity / Cassette Short Count
00AC -00AB	173–174	**Pointer:** Tape Buffer / Screen Scrolling
00B0 -00B1	176–177	Tape Timing Constants
00B2 -00B3	178–179	**Pointer:** Start of Tape Buffer
00B4	180	RS-232 Out Bit Count / Cassette Temp
00B5	181	RS-232 Next Bit to Send / Tape EOT Flag
00B6	182	RS-232 Out Byte Buffer
00B7	183	Length of Current File Name
00B8	184	Current Logical File Number
00B9	185	Current Secondary Address
00BA	186	Current Device Number
00BB -00BC	187–188	**Pointer:** Current File Name
00BD	189	RS-232 Out Parity / Cassette Temp
00BE	190	Cassette Read/Write Block Count
00BF	191	Serial Word Buffer
00C0	192	Tape Monitor Interlock
00C1 -00C2	193–194	I/O Start Address
00C3 -00C3	195–196	Tape Load Temps
00C5	197	Last Key Pressed (255 = none)
00C6	198	Keystroke Buffer Count
00CC	204	0 = Cursor Enable 1 = Cursor Disable
00CD	205	Cursor Timing Countdown
00CE	206	Character Under Cursor
00CF	207	Last Cursor Blink: ON/OFF
00D0	208	Input from Screen / from Keyboard
00D1 -00D2	209–210	Current Physical Screen Line Address
00D3	211	Position of Cursor on Line
00D4	212	Quote Mode (0 = off 1 = on)
00D5	213	Current Physical Screen Line Length
00D6	214	Line Cursor is on (0–24)
00D7	215	Last Inkey/Checksum/Buffer
00D8	216	Number of Inserts Outstanding
00D9 -00F2	217–242	Screen Line Link Table / Line Wrap Table
00F3 -00F4	243–244	**Pointer:** Current Screen Color Map Start
00F5 -00F6	245–246	**Vector:** Keyboard Decode Table
00F7 -00F8	247–248	**Pointer:** RS-232 Input Buffer
00F9 -00FA	249–250	**Pointer:** RS-232 Output Buffer
00FB -00FD	251–253	Free Memory (zeroed by NEW and Chain)
00FE	254	Free Memory
0100 -01FF	256–511	Microprocessor Stack Area
0200 -0258	512–600	System Input Buffer
0259 -0262	601–610	Kernal Table: Active Logical File Numbers
0263 -026C	611–620	Kernal Table: Device Number for each File
026D -0276	621–630	Kernal Table: Secondary Address for each File
0277 -0280	631–640	Keyboard Buffer
0285	645	**Flag:** Kernal Variable for IEEE Timeout
0286	646	Current Pencolor
0287	647	Current Color Under Cursor (Background Color)
0288	648	Top of Screen Memory Page
0289	649	Size of Keyboard Buffer
028A	650	Repeat Enable 128 = repeat any key after approx 1/2 second
028B	651	Repeat Speed Counter
028C	652	Repeat Delay Counter
028D	653	Special Keys (0 = none 1 = Shift 2 = Commodore Key 4 = Control Key)
028E	654	Last Keyboard Shift Pattern
028F -028E	655–656	**Vector:** Keyboard Table Setup
0291	657	**Flag:** 0 = Disable Shift Key, 128 = Enable Shift Key
0292	658	**Flag:** Auto Scroll Down, 0 = on
0293	659	RS-232: 6551 Control Register Image
0294	660	RS-232: 6551 Command Register Image
0295 -0296	661–662	RS-232: Non-Standard BPS (time/2-100) USA
0297	663	RS-232: 6551 Status Register Image
0298	664	RS-232: Number of bits left to send
0299 -029A	665–666	RS-232 Baud Rate: full bit time (micro seconds)
029B	667	RS-232 Index to End of Input Buffer
029C	668	RS-232 Start of Input Buffer (page)
029D	669	RS-232 Start of Output Buffer (page)
029E	670	RS-232 Index to End of Output Buffer
029F -02A0	671–672	Holds IRQ Vector during Tape I/O
02A1	673	RS-232 Enables
02A2	674	TOD Sense during Cassette I/O
02A3	675	Temp Storage for Cassette Read
02A4	676	Temp D1IRQ Indicator for Cassette Read
02A5	677	Temp for Line Index
02A6	678	PAL/NTSC Flag, 0 – NTSC / 1 = PAL
02A7 -0313	679–787	UNUSED! 108 Bytes

0314 -0315	788–789	IRQ Vector			
0316 -0317	790–791	BRK Instruction Vector			
0318 -0319	792–793	NMI Vector			
031A -031B	794–795	OPEN Vector			
031C -031D	796–797	CLOSE Vector			
031E -031F	798–799	CHKIN Vector			
0320 -0321	780–781	CHKOUT Vector			
0322 -0323	782–783	CLRCHN Vector			
0324 -0325	784–785	CHRIN Vector			
0326 -0327	786–787	CHROUT Vector			
0328 -0329	808–809	STOP Vector (Scan for STOP Key pressed)			
032A -032B	810–811	GETIN Vector			
032C -032D	812–813	CLALL Vector			
032E -032F	814–815	User Defined Vector			
0330 -0331	816–817	LOAD Vector			
0332 -0333	818–819	SAVE Vector			
0334 -033B	820–827	UNUSED! 7 Bytes			
033C -03FB	828–1019	Disk / Cassette Buffer			
0400 -07E7	1024–2023	Text Screen Memory			
07E8 -07F7	2024–2039	Free Memory			
07F8 -07FF	2040–2047	Sprite Pointers (not applicable normally)			
0801	2049	BASIC program 'sys 2063'			
0812 -0813	2066–2067	Top Address Space available on power-up (only used once)			
07E8 -0811	2024–2065	UNUSED (by COMAL) 22 Bytes			
0814 -0ACA	2068–2762	Start of COMAL Keyword Table Format 1 Byte Length of word followed by Command Word (CBM Format)			
10E1	4321	Linefeed After Carrage Return if not zero (0)			
10E5 -10E6	4325–4326	Old IRQ Vector			
10FC	4348	Output Location 0 = screen 1 = printer – see also 152 ($0098)			
1105	4357	Routine to Send Carrage Return (and Linefeed if necessary)			
19D0	6608	SYS to this location to call the Error Number in Loc 43 ($002b)			
2CEC -2CF9	11500–11513	Code to Reset DIM Variables and High Mem Pointers			
2D55	11605	New Text IRQ			
2E7E	11902	New Graphics NMI			
2E94	11924	New Graphics IRQ			
2EAF	11951	New Text NMI			
2EE2	12002	Number of Border Color used by RUN/STOP RESTORE			
2EE7	12007	Number of Background Color used by RUN/STOP RESTORE			
2EEC	12012	Number of Pencolor used by RUN/STOP RESTORE			
		COMAL starts here			
2F04 -2F39	12036–12089	Setup New Interrupt Vectors: Hardware IRQ Vector to 11605 ($2D55) and NMI Vector to 11951 ($2EAF)			
2F3A -2F50	12090–12112	Copy BASIC ROM to hidden RAM underneath			
2F51 -2F54	12113–12116	Switch BASIC ROM Out			
2F55 -2F59	12117–12121	Set Background Color to Blue			
2F5A -2F5E	12122–12126	Set Border Color to Light Blue			
2F5F -2F7F	12127–12159	print 'initial greeting screen'			
30FF	12543	Prints the '9902' portion of 9902 Bytes Free			
3103	12547	General Print Message Routine use to print greeting screen. Uses 117,118			
		as Indirect Pointers to ASCII Bytes of text to print. Message ends with a $00 (hex)			
6A77 -6A78	27255–27256	X Coordinate of Turtle			
6A7A	27258	Turtle Size			
6A7C	27260	Y Coordinate of Turtle			
6A7D	27261	Type of Graphics Screen now in use – Hi-Res (0) or Multi-Color (1)			
6A8C	27276	Sprite on or off bits			
6A8D -6A8E	27277–27278	Heading of Turtle			
6A9F	27295	Turtle State – Visible (1) or Invisible (0)			
6AC5	27333	Turtle Pen State – Down (1) or Up (0)			
8753 -894F	34643–35151	Logon Message / Tokenized Display Line last entered			
8835	34869	Text entered in Quote Mode			
884B	34891	ASCII (PETASCII) Display Line last entered			
8951 -B000	35153–45056	COMAL Program Work Space			
B000	45056	Top of Programing Space			
B001 -BFFF	45057–49151	BASIC Routines copied to RAM underneath (Math, Input, etc.)			
B391	45969	Fix to Float			
B7F7	47095	Float to Fix			
B853	47187	Perform (subtract)			
B86A	45290	Perform (add)			
B9EA	47594	Perform (log)			
BA2B	47659	Perform (multiply)			
BAFE	47870	Divide by 10			
BB12	47890	Perform (divide)			
BBA2	48034	Memory to Floating Point Accumulator *1			
BBFC	48124	Move Floating Point Accumulator *2 to *1			
BC0C	48140	Move Floating Point Accumulator *1 to *2			
BC39	48185	Perform (sgn)			
BC58	48216	Perform (abs)			
BC5B	48219	Compare Floating Point Accumulator *1 to memory			
BC9B	48283	Float to Fix			
BCCC	48332	Perform (INT)			
BDDD	48605	Float to ASCII			
BFB4	49076	Perform (Negative)			
BFED	49133	Perform (EXP)			
C000 -C03F	49152–49215	Sprite Image 0	C040 -C07F	49216–49279	Sprite Image 1
C080 -C0BF	49280–49343	Sprite Image 2	C0C0 -C0FF	49344–49407	Sprite Image 3
C100 -C13F	49408–49471	Sprite Image 4	C140 -C17F	49472–49535	Sprite Image 5
C180 -C1BF	49536–49599	Sprite Image 6	C1C0 -C1FF	49600–49663	Sprite Image 7
C200 -C23F	49664–49727	Sprite Image 8	C240 -C27F	49728–49791	Sprite Image 9
C280 -C2BF	49792–49855	Sprite Image 10	C2C0 -C2FF	49856–49919	Sprite Image 11
C300 -C33F	49920–49983	Sprite Image 12	C340 -C37F	49984–50047	Sprite Image 13
C380 -C3BF	50048–50111	Sprite Image 14	C3C0 -C3FF	50112–50175	Sprite Image 15
C400 -C43F	50176–50239	Sprite Image 16	C440 -C47F	50240–50303	Sprite Image 17
C480 -C4BF	50304–50367	Sprite Image 18	C4C0 -C4FF	50368–50431	Sprite Image 19
C500 -C53F	50432–50495	Sprite Image 20	C540 -C57F	50496–50559	Sprite Image 21
C580 -C5BF	50560–50623	Sprite Image 22	C5C0 -C5FF	50624–50687	Sprite Image 23
C600 -C63F	50688–50751	Sprite Image 24	C640 -C67F	50752–50815	Sprite Image 25
C680 -C6BF	50816–50879	Sprite Image 26	C6C0 -C6FF	50880–50943	Sprite Image 27
C700 -C73F	50944–51007	Sprite Image 28	C740 -C77F	51008–51071	Sprite Image 29
C780 -C7BF	51072–51135	Sprite Image 30	C7C0 -C7FF	51136–51199	Sprite Image 31
C800 -C83F	51200–51263	Sprite Image 32	C840 -C87F	51264–51327	Sprite Image 33
C880 -C8BF	51328–51391	Sprite Image 34	C8C0 -C8FF	51392–51455	Sprite Image 35
C900 -C93F	51456–51519	Sprite Image 36	C940 -C97F	51520–51583	Sprite Image 37
C980 -C9BF	51584–51647	Sprite Image 38	C9C0 -C9FF	51648–51711	Sprite Image 39
CA00 -CA3F	51712–51775	Sprite Image 40	CA40 -CA7F	51776–51839	Sprite Image 41
CA80 -CABF	51840–51903	Sprite Image 42	CAC0 -CAFF	51904–51967	Sprite Image 43
CB00 -CB3F	51968–52031	Sprite Image 44	CB40 -CB7F	52032–52095	Sprite Image 45
CB80 -CBBF	52096–52159	Sprite Image 46	CBC0 -CBFF	52160–52223	Sprite Image 47
CC00 -CC3F	52224–52287	Sprite Image 48	CC40 -CC7F	52288–52351	Sprite Image 49
CC80 -CCBF	52352–52415	Sprite Image 50	CCC0 -CCFF	52416–52479	Sprite Image 51
CD00 -CD3F	52480–52543	Sprite Image 52	CD40 -CD7F	52544–52607	Sprite Image 53
CD80 -CDBF	52608–52671	Sprite Image 54	CDC0 -CDFF	52672–52735	Sprite Image 55
CE00 -CFFF	52736–53247	RS-232 Buffer (512 Bytes)			
D000	53248	Start VIC Chip – refer to Programmers Reference Guide page 321			
D000	54238	Start of first Character Generator ROM (UPPER/GRAPHICS)			
D400	54272	Start SID Chip – refer to Programmers Reference Guide page 324			
D800	55296	Start of Screen Character Colors & Graphics Screen (hi-res Color Map under the I/O)			
		Start of second Character Generator ROM (lower/UPPER)			
DBC0 -DBFF	56256–56319	Start of Turtles Current Image – just a guess			
DC00	56320	Start CIA1 Chip (Keyboard CIA Chip) refer to Programmers Reference Guide page 328			
DC01	56321	Joystick & Button – Port 1			
DC08 -DC0B	56328–56331	Hardware Clock / Timer			
DC0D	56333	Poke ,1 – Disable Timer A Interrupt // Poke ,129 Enable			
DC0F	56335	Part of Clock / Timer			
E000 -FFFF	57344–65535	Start of Bit map for Graphics Screen			

Printer Control Characters

CHR$ values are sent to printer with Secondary Addr 0 or 1. Not all codes are implemented on all printers

CHR$	Operation	CHR$	Operation	CHR$	Operation
1	Begin double-width (enhanced) character mode	**14**	Begin double-width character mode	**19**	Set top of page
129	End double-width character mode	**15**	End double-width character mode	**147**	Feed to top of next page
8	Begin dot-programmable graphic mode	**16**	Tab to position in next 2 characters	**26**	Repeat graphics data
10	Line Feed	**17**	Switch to upper/lower case character set	**27**	Move to specified dot position
13	'Carriage Return' (automatic Line Feed on CBM printers)	**145**	Switch to upper case/graphics character set	**29**	Skip to next format field
141	Carriage Return without Line Feed	**18**	Begin reverse character mode	**160**	Shifted Space is necessary for leading spaces
		146	End reverse character mode	**254**	Output Programmable Character

Commodore Dot-Matrix Printer Format Characters

Format Char	Format Supplied	Data Supplied	Output Result	Description
9	99999.99 .99 99.99	3.14159 3.14159 23	3.14 .14 23.00	Specifies numeric field, leading zeros suppressed
z	zzzzz.zz	3.14159	00003.14	Specifies numeric field, leading zeros printed
.				Decimal point. Used to align data
$	$$$$.99	129.95	$129.95	Specifies numeric field with a $ sign printed preceding data
s	s999.99 s$$$$.99	-273.6 129.95	-273.60 +$129.95	Prints sign of value as first character in field
–	$999.99– s999.99– s$999.99–	-129.95 -273.6 129.95	$129.95– -273.60– +$129.95	Prints trailing sign if negative
a	aaaaaa aaa	String String	String Str	Specifies a left-justified alpha field
ƀ				Space or blank. Use spaces to seperate fields
r	r?aaaa 999	over 100	?over 100	Allows format-string characters to be printed

Letter Quality Printer Command Summary

Commands are for the StarWriter F10 printer. Most letter-quality printers are similar. Note: ESC is escape, or chr$(27).

Command Format	Description	Command Format	Description
chr$(12)	Form Feed	**ESC Pnn**	Feed paper to line nn
chr$(8)	Backspace	**ESC A**	Alternate Ribbon Colour
ESC Lnn	Line feed spacing	**ESC B**	Normal Ribbon Colour
ESC chr$(10)	Backwards Line Feed	**ESC U**	Half Line Feed
ESC 9	Set Left Margin	**ESC D**	Half Backwards Line Feed
ESC Enn	Set horizontal spacing to nn/120	**ESC 1**	Set Horizontal Tab at Current position
ESC 2	Clear all horizontal tabs	**ESC Hnnn**	Move Carriage nnn spaces horizontally
ESC 8	Clear one Horizontal tab at current position	**ESC Vnnn**	Line feed of nnn/48 inches
ESC (t1,t2,...ff	Sets horizontal tabs at t1, t2, etc.	**ESC Fnn**	Set number of lines per page
ESC)t1,t2,...ff	Clears horizontal tabs at t1, t2, etc.	**ESC N**	Ignore auto-spacing on next character
ESC Cnn	Move to Column nn		

Greek Alphabet

Dot Matrix CHR$ Values						Letter	Upper Case	Lower Case	Roman Equiv.	Common Unit
14	17	10	4	26	1	Alpha	A	α	A	Area, Angles, Coefficients
0	1	62	80	42	4	Beta	B	β	B	Angles, Coefficients, Flux Density, Transistor Amplification Factor
0	64	54	9	54	64	Gamma	Γ	γ	G	Specific Gravity, Conductivity, Micrograms
0	22	41	41	6	0	Delta	Δ	δ	D	Density, Variation
0	10	21	21	17	2	Epsilon	E	ε	E	Natural Logarithm Base (e'' = 2.1242657)
0	64	44	50	35	64	Zeta	Z	ζ	Z	Coefficients, Coordinates, Impedance
0	64	48	65	62	0	Eta	H	η	H	Efficiency, Hysteresis Coefficient
0	62	73	73	62	0	Theta	Θ	θ	V	Phase Angle, Temperature
0	0	30	1	2	0	Iota	I	ι	I	
17	14	4	8	30	17	Kappa	K	κ	K	Dielectric Constant, Susceptibility
65	66	52	12	2	1	Lambda	Λ	λ	L	Wavelength
1	126	32	32	120	4	Mu	M	μ	M	Amplification Factor, micro (10^{-6}), Permeability
0	16	12	3	4	24	Nu	N	ν	N	Reluctivity
0	66	53	41	65	0	Xi	Ξ	ξ	Y	
0	6	9	17	18	12	Omicron	O	ο	O	
0	9	30	16	30	33	Pi	Π	π	P	3.1415926
0	62	73	72	48	0	Rho	P	ρ	R	Resistivity
6	9	9	14	8	8	Sigma	Σ	ς	S	Summation
99	85	73	65	65	99	Capital Sigma				
0	8	16	30	17	16	Tau	T	τ	T	Time Constant
8	6	1	1	18	12	Upsilon	Y	υ	U	
48	73	14	24	40	48	Phi	Φ	ϕ	F	Angles, Magnetic Flux
34	36	24	22	33	65	Chi	X	χ	X	
112	9	126	8	48	64	Psi	Ψ	ψ	W	Dielectric Flux, Phase Difference
0	6	9	2	9	6	Omega	Ω	ω	Q	Ohms, Angular Velocity
25	38	64	64	38	25	Capital Omega				

Wordprocessing Reference Guide

Function	Superscript Control = RVS Key	EasyScript 64 Control = F1 Key	PaperClip Control = PET/CBM:RVS, 64:CTRL	Speedscript 64 Control = CTRL Key	WordPro Control = RVS Key	WordPro 64 Control = CBM Key
Restart	Control CLR	Control CLR				
Exit to BASIC	Control STOP	Control STOP	Control X		Control Shift Q	Control Q
TOGGLE MODES	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Capitals	ESC or Control Shift/C	Control Shift/C	↑		\	£
Decimal	Control .	F6			Control N	
Insert	Control I	Control I	Shift Ctrl (64:CBM Key)	Control I	Shift Control	Control I
Sound	Control *	Control *			Control \	
LINE Mode						F1
Forced Space Mode						Control –
CURSOR POSITIONING	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Scroll Right	CRSR Right	CRSR Right	CRSR Right	CRSR Right	CRSR Right	
Scroll Left	CRSR Left	CRSR Left	CRSR Left	CRSR Left	CRSR Left	
Scroll Down	CRSR Down	CRSR Down	CRSR Down	CRSR Down	CRSR Down	
Rapid Scroll Down			Control CRSR Down	Control CRSR Down	Control CRSR Down	
Scroll Up	CRSR Up	CRSR Up	CRSR Up	CRSR Up	CRSR Up	
Rapid Scroll Up			Control CRSR Up	CRSR Up	CRSR Up	
Up a Line	←	←				
Next Screen	Control Space	Control Space				
Previous Screen	Control Shift/Space	Control Shift/Space				
Next Word				F1		
Previous Word				F2		
Next Sentence				F3		
Previous Sentence				F4		
Next Paragraph				F5		
Previous Paragraph				F6		
Beginning of File	CLR	CLR	HOME twice		HOME twice	
Home Position	HOME	HOME	HOME		HOME	HOME
End of Text	Control G E or 0	Control G E	Shift RUN/STOP	Control Z		
Goto Line x	Control G	Control G			Control G	Control G
Goto Maximum Line Number	Control G 999	Control G 999				
Pan Up	Control CRSR Up	Control CRSR Up				
Pan Down	Control CRSR Down	Control CRSR Down				
Pan Left	Control CRSR Left	Control CRSR Left				
Pan Right	Control CRSR Right	Control CRSR Right				
Stop Panning	STOP	STOP				
Speed Panning	Shift	Shift				
Highlight Panning Cursor	hold Space					
Pause Panning	tap Space	tap Space				
TEXT	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Change Line Length	Control CLR	Control CLR	Control Shift L			
Reformat Paragraph						Control R
Delete Line	Control DEL	Control DEL	Control –		Control DEL	
Insert Line	Control INST	Control INST	Control +		Control INST	
Insert Multiple Lines			Control I			
Delete Text	Control D	Control D	Control D	Control D		
Erase All	Control E A	Control E A	Control E	Shift CLR	Control E A	Control E A
Erase Remainder	Control E R	Control E R			Control E R	Control E R
Erase Paragraph	Control E P	Control E P		Control D P or E P		Control D P
Erase Sentence	Control E S	Control E S		Control D S or E S	Control D S	Control D S
Erase Word				Control D W or E W	Control D W	Control D W
Erase Delete Buffer				Control K		
Retrieve Buffer Contents				Control R		
Set Range	Control R	Control R	Control R		Control R	Control J
Transfer Range	Control T	Control X	Control T		Control T	Control T
Copy Range	Control A	Control A	Control C		Control L	
Erase Range			Control E		Control E L	
Append Characters					Control V	Control V
Append Lines					Control A	Control A
Switch Text Space					Control X	Control X
Set Column			Control Shift C			
Move Column			Control Shift M			
Delete Column			Control Shift D			
Erase Column			Control Shift E			
Shift Column			Control Shift S			
Insert Space Before Column			Control Shift I			
Repeat Column			Control Shift R			
Add Numbers in Column			Control =		Control =	
Sort Column			Control Shift A			
Set Sort Delimiters			Control Shift Q			
Set Delimiter Column			Control Shift W			
Add Row Using Delimiters			Control Shift H			
Modify Hunt/Search & Replace Text	Control M	Control S	Control F	Control Shift H	Control M S or R	
Hunt or Find Local	Control H L	Control H L	Control F or H	Control H	Control H or F L	Control H or F L
Hunt or Find Global	Control H G	Control H M			Control F G	
Hunt C	Control H C					
Display Old Search & Replace			Shift RUN/STOP			
Search & Replace Local	Control @ L	Control @ L	Control @		Control @ L	Control @ L
Search & Replace Global	Control @ G	Control @ M	Control @		Control @ G	Control @ G
Set Phrase			Control P			
Move Phrase			Control M			
Kill Phrase			Control K			
Toggle Case	Control U	Control U		Control A		
Toggle Case in Phrase			Control Shift K			
Transpose Characters				Control X		
Change Border Colour			F2	Control B		
Change Background Colour			F4	Control B		
Change Character Colour			F6	Control L		
Copy Text to Status Line	Shift Control		RUN/STOP		\	
Copy NX Filename to Status Line			Shift RUN/STOP		HOME	HOME
Read Stored Filename						F5
Display Available Memory				Control =		
Automatic Optional Hyphen	Control –	Control –	Control :		Control –	Shift –
Forced Space	Shift Space	Shift Space	Shift Space		Shift Space	Shift Space
Breakpoint (soft Space)			Control 1			

TABS	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Set Decimal Point			Control .			
Set Decimal Tabs	Control .	Control .	Control N		Control N	
Set Horizontal Tab	Control S H	Control T H	Shift CLR		Control S	Control S
Clear Horizontal Tab	Control C H	Control C H	Shift CLR		Control C	Control C
Tab 5 Spaces				RUN/STOP		
Set Vertical Tab	Control S V	Control T V				
Clear Vertical Tab	Control C V	Control C V				
Set Graphic Tab						Control £
Goto Next Horizontal Tab	TAB (or Shift >)	F7	TAB or RUN/STOP		TAB or ←	←
Goto Next Vertical Tab	Shift TAB (or Shift <)	F8				
Display Horizontal Tab positions	Control P	Control P				
Clear All Tabs			Control CLR		Control K	Control K
Clear All Horizontal Tabs	Control K H	Control Z H				
Clear All Vertical Tabs	Control K V	Control Z V				

FILES	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Enter FILE Mode					Shift CLR	F7 or CLR (F1 cancels)
Insert or Merge Files	Set Insert Mode, Load	Set Insert Mode, Load	Control A		Shift CLR I	F7 or Shift CLR I
Load PRG Text File	Control L	Control L	Control L	F7	Shift CLR R	F7 or Shift CLR R
Load SEQ Text File	Control L Control	Control L Control	Control J			F8
Load Printer Interface File			Control W			Control P
Save PRG Text File	Control F	Control F	Control S	F8	Shift CLR M	F7 or Shift CLR M
Save SEQ Text File	Control F Control	Control F Control	Control Z			
Verify Data File			Control U	Control V		
Save Range	Control Shift F	Control Shift F	Control Q		Shift CLR M R	F7 or Shift CLR M R
Read Screen from Cursor						F3
Copy Global/Linked Files	Control Q	Control Q	Control G		Control *	Control *
Scan loaded Directory names	Shift Control					
Disk Command Mode	Control >	F4	Control >	Control ↑	Control . or >	Control .
Display Directory	$0 or $1	$0 or $1	Control 0, 1, 2	Control 4		F3
Load Directory to Text	+$0 or +$1	+$0 or +$1		$0, $1	Control 0, 1, 2	
Display Disk Status	RETURN	RETURN	Control <	RETURN	Control ,	Control ,
Initialize Drive(s)	i0 or i1	i0 or i1	i0 or i1	i0 or i1	RUN/STOP 0 or 1	i0 or i1

All other disk commands are entered in CBM DOS Command Channel format (ie c = Copy, d = Duplicate, n = New, r = Rename, s = Scratch, v = Validate).

FILL FILES	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Set Fill File Name			Control Shift Z			
Variable Block	Control B	Control B	Control B		Control B	Control B
Variable Block Separator					Control Z	Control Z
Measured Variable Block		Ctrl B, CRSR Left, Ctrl M				
Fill Next Variable Block			Control Shift B		Control TAB	Control M
Fill Blocks from Cursor on	Control V	Control V				
Fill All Variable Blocks			Control Shift V		Control I	F4 (1st set) or F6
Clear Variable Blocks	Control Shift V	Control Shift V	Control Shift N		Control ↑	F2
Find Next Variable Block	Control TAB or Shift >	Control F7	Control Shift F		Control TAB	
Reset Data Pointer					Control HOME	
Close Fill File	Control HOME	Control HOME				

OUTPUT FORMAT	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Format Command Indicator	Control / (✓)	F3 (■)	Control \ or £ (✓)	Control £	Control / (✓)	Control / (✓)
Format Command Separator	:	:	:		:	↑
Text following Format Commands	;	;	;			
Justification On, Off	✓ju1, ju0	■ ju1, ju0	✓ju1, ju0		✓ju1, ju0	✓ju1, ju0
Centering On, Off	✓cn1, cn0	■ cn1, cn0	✓cn1, cn0	c	✓cn1, cn0	✓cn1, cn0
Right Alignment On, Off	✓ra1, ra0	■ ra1, ra0	✓ra1, ra0		✓ra1, ra0	✓ra1, ra0
Linefeeds On, Off	✓lf1, lf0				✓lf1, lf0	
Left Margin	✓lm	■ lm	✓lm	l	✓lm	✓lm
Add to Left Margin			✓lm+			
Subtract from Left Margin			✓lm–			
Right Margin	✓rm	■ rm	✓rm	r	✓rm	✓rm
Edge Right				e		
Add to Right Margin			✓rm+			
Subtract from Right Margin			✓rm–			
Release Left Margin Left	✓ma	■ ma	✓ma–		✓ma	✓ma–
Release Left Margin Right			✓ma+			✓ma+
Auto Indent Paragraphs Right			✓ai+			
Auto Indent Paragraphs Left			✓ai–			
Offset from Column 1 on Printer	✓of	■ of				✓mo
Double Column Width						✓dc (1-160)
Total Lines per Page (Paper Length)	✓pp	■ pl	✓pp		✓pp	✓pp
Text Lines per Page (Text Length)	✓pg	■ tl	✓pg		✓pg	✓pg
Line Spacing	✓sp	■ sp	✓sp		✓sp	✓sp
Vertical Positioning	✓vp	■ vp	✓vp	t	✓vp	✓vp
Bottom Margin				b		
Advance Lines	✓ln	■ ln	✓ln		✓ln	✓ln
Pause Output	✓ps	■ ps	✓ps	w	✓ps	✓ps
Force Paging	✓fp0	■ fp0	✓fp		✓fp	✓fp
Force Paging within N Lines	✓fpN	■ fpN	✓fpN		✓fpN	✓fpN
# of List Data Fields						✓ld
Next Linked File	✓nx:filename	■ nx:filename	✓nx:filename		✓nx:filename	✓nx:filename
Non-Specific Global File Link			✓lk			
External File Link			✓ex:			
Open Table of Contents File			✓tf:filename			
Add to Table of Contents File			✓tb:			
Lines per Inch (form advance)	✓fa	■ lp	✓ls		✓fa	✓fa
Characters per Inch (pitch)	✓pt	■ pt	✓pt		✓pt	✓pt
Comment	✓cm	■ nb	✓cm		✓cm:	✓cm:
Heading	✓hdxx:text,,	■ hdxx:text,,	✓hdxx:text,,	h	✓hdxx:text,,	✓hdxx:text,,
Alternate Heading						✓hdxx;text,,
Footing	✓ftxx:text,,	■ ftxx:text,,	✓ftxx:text,,	f	✓ftxx:text,,	✓ftxx:text,,
Alternate Footing						✓ftxx;text,,
Set Page Number	✓p#	■ p#	✓p#		✓p#	✓p#
Output Page Number	Control # (in hd/ft)	Control # (in hd/ft)	Control # (in hd/ft)	Control £ #	<> (in hd/ft)	Shift £
Heading/Footing Left Margin	✓hl	■ hl	✓hl		✓hl	✓hl
Heading/Footing Right Margin	✓hr	■ hr	✓hr		✓hr	✓hr
Unlock Header Margins			✓ml0			
Lock Header Margins			✓ml1			
Printer Command						✓pc
Send True ASCII	Control 0-9	Control 0-9	Control : 1-9	a	Control 0-9	Control 0-9
Define Character as ASCII Value	✓1-9 = N	■ 1-9 = N	✓1-9 = N	Control £ 1-9 = N	✓0-9 = N	✓0-9 = N

OUTPUT	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Select Default Output			Control Shift O			
Set Disk Device Number			Control $			
Set Printer Device Number			Control #			
Select Output Options	Control O +	Control O +	Control O +	Control P	Control O +	F5 or Control O +
Continuous Print	C	C			C	C
Non-Continuous (sheets)						N
Device Number	D					
Fill File to be used	F	F			S	
Fill Using List Data					L	L
Linked or Global File	G	L			G	G
Global Restart						R
Map Mode						M
Odd Mode (odd # pages)						O
Even Mode (even # pages)						E
Number of Copies	X	X			X	X
Output to Printer	P	P			Default	
Output to Video	V	V	Control V		V	V
Output to SEQ file	S	S			D	
Speed up Video Output	Hold down Shift	Hold down Shift				
Pause Video Output	Tap space	Tap space				
Stop Output	STOP	STOP				
Continue Output	C	C				
Toggle Video/Printer Output	V/P	V/P				
Toggle Continuous/Non-Continuous	Shift P	Shift P				
Toggle Map/Video Mode						A

BACKGROUND PRINTING	Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Start Background Printing	Control X				Control P (file "dp")	
Resume after Page Break	Z (X for non-8032)					
Stop Background Printing	Control Shift X				Control P	

PRINTER CONTROL CHARACTERS			Superscript	EasyScript 64	PaperClip	Speedscript 64	WordPro	WordPro 64
Letter Quality	MX80	CBM						
Underline ON	Enhance ON	Enhance ON	Control [	Control [	Control [	Control £ U	Control [	Control [
Underline OFF	Enhance OFF	Enhance OFF	Control]	Control]	Control]	Control £ U	Control]	Control]
Bold ON	Emphasise ON	Reverse ON	Control (	Control (	Control (		Control 8	Control (
Bold OFF	Emphasise OFF	Reverse OFF	Control)	Control)	Control)		Control 9	Control)
Shadow ON	Double print ON	n/a	Control &					
Shadow OFF	Double print OFF	n/a	Control					
Print Red	Condense ON	n/a	Control !	Control Shift (				
Print Black	Condense OFF	n/a	Control "	Control Shift)				
Single Superscript	n/a	n/a	Control '		Control 4		Control 4	Control 6
Superscript Begin	n/a	n/a			Control 7			
Superscript End	n/a	n/a			Control 8			
Single Subscript	n/a	n/a	Control ,		Control 6		Control 6	Control 4
Subscript Begin	n/a	n/a			Control 9			
Subscript End	n/a	n/a			Control /			
Bold ON	n/a	n/a	Control ;					
Bold OFF	n/a	n/a	Control :					
Special Character	Special Character	Special Character			Control ;			

Spreadsheet Commands

Commands shown are for the CalResult spreadsheet, but most spreadsheet programs use similar syntax.

System Commands:	Description
B	Blank: Cancel Contents of Cell Under Cursor
L	Leave: Title, Split-Screen, Window
O	Order of Recalculation (Row or Column)
Q	Quit Program
R	Recalculate: Automatic or Manual
–	Automatic Repetition of Characters at Cell Under Cursor

E: Edit Command	Description
E C	Copy Data Area to another Data Area
E D	Delete Row or Column
E G	Graphics: Histogram instead of Values
E I	Insert Row or Column
E M	Move Data Area to another Data Area
E P	Print Worksheet or User-Defined Format
E R	Replicate Data Area to other Data Areas
E S	Split Screen (Horizontally or Vertically)
E T	Title: Protects a Title in the Left Column
E W	Insert Window on Screen

F: Format Command	Description
F C	Select Colour
F G	Global Cell: Sets global format Global: Clears all Formats to CalcResult's normal power-up mode (labels left, values right and maximum precision)
F M	Maximum Precision display mode
F I	Integer display mode
F $	Two Decimal display mode
F L	Sets Contents at Left
F R	Sets Contents at Right
F *	Replaces Integer Number digits with stars (always left justified)

P: Page Command	Description
P A	Add Pages, checking that label and formula match
P C	Copy one Page to another
P D	Delete Page from Work Area
P E	Erase Work Area
P G	Get Page from Work Area
P N	Negate: Change Signs (+ and –) in one Page
P P	Put 2nd Page from Work Area (to get extra memory)
P R	Renumber Page
P +	Add Pages, reading Values and Formulae only

G: Global Command	Description
G C	Sets Global Column Width, except in Protected Title-Column
G F	Set Format in all Cells
G R	Recalculate Pages by moving the highest column in one Page to the Alpha Column in the Next Page

D: Disk Command	Description
D B	Backup Drive 0 to Drive 1
D C	Catalog of Drive 1
D D	Save and Load DIF-files
D E	Erase File on Drive 1
D I	Initialize Drives 0 and 1
D L	Load File from Disk to Work Area
D N	New Disk (formatted in Drive 1)
D S	Save Work Area to Drive 1
D U	User Register: Contains language for Help screens, type of printer, paper format, etc. Type of Printer: 1 = 8023P 2 = 4022 4 = ASCII 3 = 8024, 8026, 8027, 8028, 8026b
D V	Load a VisiCalc-File

Commodore +4: 3+1 Software Reference Guide

Word Processor

Special Keys:

INST/DEL	Insert/delete character	**CTRL 9**	Set reverse video for formatting instructions
HOME	Move cursor to top line of text	**CTRL 0**	Turn off reverse video
CLR	Move cursor to bottom line of text	**C= C**	Enter command mode
RETURN	Terminate a paragraph	**F1 or C= L**	Move cursor to left margin
SHIFT RETURN	Move cursor to left margin of next line	**F2 or C= R**	Move cursor to column 41
SHIFT =	Tab key	**C= Q**	Repeat previous keystroke
CTRL =	Set a tab	**C= @**	Replace line deleted by a RETURN

Commands: All commands are initiated with C= C

CA	Display disk directory (CAtalog)	**DL**	Delete a Line of text	**PR**	Saves current document to disk with name "..tw" then prints it
CB	Create a Block	**EP**	Erase a Pointer	**RE**	Search and Replace words or phrases
CM	Clear Memory	**IB**	Insert a Block created with CB	**SF**	Save File to disk
CP	Clear Pointers	**ID**	Initialize Disk	**SP**	Set a Pointer
CT	Clear Tabs	**IL**	Insert a Line of text	**SR**	Search for a word or phrase
DB	Delete Block	**LF**	Load a File from disk	***P**	Print document
DF	Delete a disk File	**MF**	Merge a File from disk into text		

Formatting Instructions: (enter in lowercase)

ASC	Send an ASCII character to the printer	**OTHER**	Used for non-Commodore printers (standard ASCII)
CENTER	Center the text on the current line	**PAGELENn;**	Set the number of lines on a page to 'n' lines (default 60)
JUSTIFY	Right-justify text	**PAGEPAUSE**	Stops printing after each page
LINKFILE	Links documents at print time	**PAPERSIZEn;**	Sets up paper size to 'n' lines long (default 66)
LMARGn;	Set left margin to 'n' (default 0)	**PAUSE**	Stops printing until RETURN is pressed
NEXTPAGE	Forces a new page	**RMARGn;**	Sets the right margin to 'n' (default 77)
NOJUSTIFY	Turns off right justification (default)	**SET#PGn;**	Sets page number to 'n'
NOWRAP	Turns off word-wrap; used for spreadsheet tables	**#PAGE**	Prints page number at bottom of each page
NO#PAGE	Turns off page numbering	**WRAPON**	Turns word-wrap on (default)

Spreadsheet

Special Keys:

Cursor Down	moves the cursor down a cell	**F1 or C= L**	moves the cursor left a cell	**C= T**	Enter text in current cell
Cursor Up	moves the cursor up a cell	**C= C**	enters command mode	**C= F**	Enter a formula in current cell
F2 or C= R	moves the cursor right a cell	**C= Q**	repeats last command	**C= N**	Enter a number in current cell

Commands: (Command mode is entered with C= C)

AUTO	Turns on automatic calculation mode	**HOME**	Moves the cursor to cell 1;1
BLKMAPr;c	Moves block of cells from cursor to 'r;c' into the Word Processor	**ID**	Initialize Disk
CA	Display disk directory	**IN**	Displays number in current cell in integer format
CCO c;	Copies column 'c' to the cursor's column	**LEFTJ**	Left justifies number in current cell
CDEL	Deletes the current column	**LF**	Load spreadsheet File from disk
CINS	Inserts a new column	**MAN**	Manual calculation mode (default)
CM	Clear memory; deletes current spreadsheet	**MAP**	Maps cell contents into the Word Processor
COLOR n;	Changes the screen colour to colour 'n' (default 0)	**OFF**	Turns off MAP mode (default)
COPY r;c	Copies cell 'r;c' to the current cell	**RCO r;**	Copies row 'r' to the current row
DF	Delete a disk file	**RDEL**	Deletes the current row
FIT r;c	Copies the formula in 'r;c' to current cell and adjusts it to reflect the new cell position	**RESET**	System reset (same as pressing RESET button)
		RIGHTJ	Right justifies number in current cell (default)
FL	Puts number in current cell in floating point format	**RINS**	Inserts a new row
FORMAT	Format a disk	**SF**	Saves current spreadsheet to disk
FRE	Freeze - locks a cell - cannot be modified until THAWed	**THAW**	Unfreezes a frozen cell
FU	Full screen display mode (default)	**TW**	To the Word Processor
GOTO r;c	Moves the cursor to cell 'r;c'	**$$**	Displays number in current cell in dollar format (two decimal places)
HA	Half screen display mode - allows simultaneous display of Word processor and spreadsheet		

Arithmetic Operators:

#	Indicates a numeric constant in formula	**DIV r1;c1 TO r2;c2**	Divides a series of numbers in a row or column
+, −, *, /	Add, Subtract, Mult, Divide	**MAX r1;c1 TO r2;c2**	Gives the largest value of the specified row or column
↑	Exponentiation	**MIN r1;c1 TO r2;c2**	Gives the smallest value of the specified row or column
EXP	Raises e (2.71828183) to a given power	**MLT r1;c1 TO r2;c2**	Multiplies all values in the given row or column
LOG	Calculates logarithm	**SUB r1;c1 TO r2;c2**	Subtracts all values in the given row or column
ABS	Absolute value	**SUM r1;c1 TO r2;c2**	Adds all values in the given row or column
ATN	Arctangent (in radians)	**r1;c1 ← r2;c2**	Moves the contents of cell 'r2;c2' to cell 'r1;c1'
COS	Cosine	**IFTRUE**	Used with ← to move the contents of a cell to another if the condition is true IFTRUE operators: **=, >, <, nte (not =), not**
SIN	Sine in radians		

File Manager

Commands: (C= C enters command mode)

CA	Display disk directory	**RV n;**	Reviews records in a file starting with record 'n' (pause with S, stop with Q)
DS f1;f2;f3	DiskSort - Sorts a disk file by specified fields (up to 3)	**PI**	Pick a range of records meeting certain criteria to create a subfile
HIGHRC n;	Specifies max record for sorts, searches, reviews, selects, reports	**SR**	Search for a record
NR	Next Record - updates current record and displays next record	**TC**	Move to the Spreadsheet
RC n;	Displays record number 'n'	**TF**	Display filename, number of records left, and the last record # entered
RESETLIST	Sets upper record limit set by HIGHRC to maximum number of records in the file	**TW**	To the Word Processor
		UD	Update Record - files displayed record; use UDn; to file under record # 'n'

Word Processor commands used with the File Manager

TF;:RC;	Indicates that the document is using File Manager data	**FLD n;**	Prints the contents of field number 'n'
RC n;	Start printed output with record number 'n'	**#RC**	Prints the record number
TTL n;	Prints the name of field number 'n'	**EOF?**	If placed at the end of a document, causes output to continue for all records in the file

Machine Language Monitor Commands

The following is a summary of typical MLM commands. Command syntax shown may vary slightly between different monitors.

Command	Syntax	Description
ASSEMBLE	.A 2000 BEQ $2010	Assemble at address $2000. Branch offsets are calculated.
BANK	.BBIN	Bank BASIC IN (Commodore 64)
	.BBOUT	Bank BASIC OUT
	.BKIN	Bank Kernal IN
	.BKOUT	Bank Kernal OUT
BREAK SET	.B 1000 00FF	Sets a break at 1000 HEX on the FF HEX occurence of the instruction at 1000.
COMPARE MEMORY	.C 1000 2000 C000	Print the locations of bytes from $1000 to $2000 that are unequal to corresponding memory at $C000.
DISASSEMBLE	.D 2000 3000	Disassemble from $2000 to $3000 (second parameter optional).
FILL	.F 1000 2000 FF	Fills memory from $1000 to $2000 with $FF.
GO	.G 1000	Execute code at $1000. Uses PC register as start address if none specified.
HUNT	.H C000 D000 'READ	Hunt for the ASCII string "READ" from $C000 to $D000.
	.H C000 D000 20 D2 FF	Hunt for the byte sequence of 20 D2 FF
INTERROGATE	.I 7000 8000	Displays memory from $7000 to $8000 with screen printable characters.
LOAD	.L "FILENAME", 08	Load file from device 8, BASIC text pointers unaltered.
MEMORY DISPLAY	.M 0000 0100	Display memory from $0000 to $0100.
NEW LOCATE	.N 1000 17FF 6000 1000 1FFF [W]	Relocate code from 1000 to $17FF at $6000, adjusting any address within $1000 to $1FFF. Use W to adjust WORD tables.
CALCULATE BRANCH OFFSET	.O 6000 5FFF FD	Calculate Branch Offset from $6000 to $5FFF (Result is $FD)
QUICK TRACE	.Q 1000	Trace code from $1000 (or PC if no address specified), disassembly suppressed.
POWER ON RESET	.P	Executes BASIC cold start
REGISTER DISPLAY	.R	Displays the PC, IRQ, Status or .P, .A, .X, .Y, and Stack Pointer.
SAVE	.S "1:FILENAME",08,7000,8000	Save to drive 1 from $7000 to $7FFF (end address -1)
TRANSFER MEMORY	.T 1000 1FFF 7000	Memory from $1000 to $1FFF is transfered to $7000
WALK CODE	.W 1000	Single step code from $1000 (or PC if no address specified) and disassemble each code executed.
EXIT TO BASIC	.X	Returns to BASIC READY mode.
	.E	In Micromon, combines .X with .K
	.K	In Micromon, restores BRK & IRQ vectors
CHANGE CHARACTER SETS	.Z	Upper Case/Graphics to Lower/Upper Case mode or vice versa.
HEX CONVERSION	.$4142	Displays Dec (16706), the ASCII characters (a b), and Binary (0100 0001 0100 0010
DECIMAL CONVERSION	.#16706	Displays Hex ($4142) followed by ASCII and Binary as above.
BINARY CONVERSION	.%0100000101000010	Displays Hex, Decimal, followed by ASCII
ASCII CONVERSION	."A	Displays Hex (41), Decimal (65), and Binary (0100 0001)
ADD	.+ 8000 7FFF	Displays the sum of the two Hex values (FFFF)
SUBTRACT	.- FFFF 7FFF	Displays the difference of the two Hex values (8000)
CHECKSUM	.& 7000 7FFF	Displays a Checksum of memory from $7000 to $7FFF

Assembler Commands

Assembler Pseudo-Ops

.BYTE	Place bytes in memory according to the operands specified
.DBYTE	Place 16-bit values in memory, stored hi order, low order (not in PAL)
.END	Ends assembly of a source file
.FIL	(.FILE in PAL) Links another source file to the current one
.LIB	Allows Library files to be inserted during assembly
.OPT	Sets options for assembly
.PAGE	Advances the listing to a new page (noy in PAL)
.SKIP	Generates blank lines in listing
.TEXT	(.ASC in PAL) Puts a string of ASCII characters in memory
.WORD	Puts 16-bit values in memory, stored low order, high order
***=**	Set program counter to a given address
=	Equate: assigns a value to a symbol
***=*+N**	Reserve N bytes for data storage

Additional PAL Pseudo-Ops

.IF	Conditional assembly pseudo-op. Follow with EXPR: and the source code to assemble if EXPR is true.
.GOTO	Transfers assembly to the line number specified.
.GTB	Go To BASIC. Exits assembly and enables the BASIC interpreter.
.STM	Symbol Table Minimum. Prevents the Symbol Table from inhabiting memory below the specified address.
.SST	Save Symbol Table
.LST	Load Symbol Table
.SYS	JSR to the specified address during assembly (either pass).

Prefix Characters

.	Indicates an assembler directive
#	Immediate Addressing mode
( )	Indirect Addressing mode
!	Forces Zero-Page Addressing mode
$	Specifies a hexadecimal value
%	Specifies a binary value
@	Specifies an octal value
'	Specifies an ASCII literal
;	Indicates that comments follow
<	Specifies the low byte of a 16-bit value.
>	Specifies the high byte of a 16-bit value

Expression Operators

+	Add values or expressions,
-	Subtract
*****	Multiply
!	Boolean OR
&	Boolean AND
↑	Boolean Exclusive OR
<	Placed to the right of an expression specifies the expression shifted left n bits. EXPR<4 would shift EXPR left 4 bits. EXPR can be 16 bits.
>	Placed to the right of an expression specifies the expression shifted right n bits. EXPR<4 would shift EXPR right 4 bits.
!	Forces Absolute Addressing

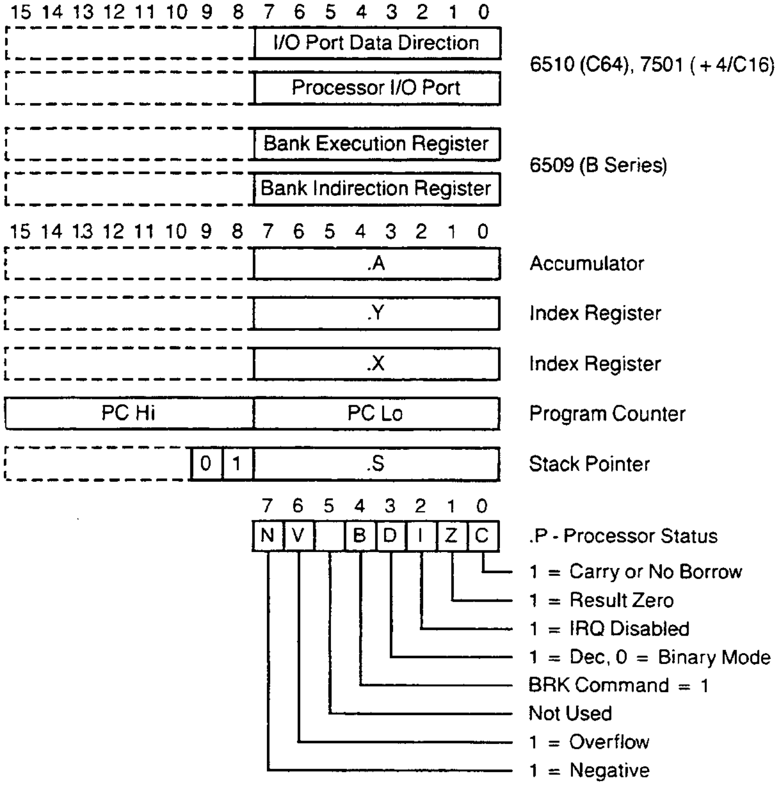
CBM .OPT Directives

ERR	Generate Error File (default)
NOE	Suppress Error File generation
LIST	Generate Listing File containing the Assembler output, including errors, comments, symbol table, etc. (default)
NOL	Suppress Listing File
MEM	Generate Memory File (default)
NOM	Suppress Memory File
GEN	Display beyond the first two bytes of a .BYTE (ie. for ASCII strings)
NOG	Show only the first two bytes of a .BYTE directive. (default)

PAL .OPT Directives

P	Print Assembly Listing
Pn	Print Assembly Listing to the previously OPENed logical file n.
P=	Print through a user routine at the address specified after the = sign (character in .A)
O	Output Object code to BASIC Arrays memory
OO	Output Object code to Origin
On	Output Object code to the previously OPENed logical file n (start address included).
O=	Output Object code through a user routine at the address specified after the = sign.
N	Null or reset .OPT directives

CPU Model



Pocket Op-Codes Chart

Mde.	IMM	ZPg	Z.X	(I.X)	(I).Y	ABS	A.X	A.Y
Byts.	2	2	2	2	2	3	3	3
ORA	09	05	15	01	11	0D	1D	19
AND	29	25	35	21	31	2D	3D	39
EOR	49	45	55	41	51	4D	5D	59
ADC	69	65	75	61	71	6D	7D	79
STA		85	95	81	91	8D	9D	99
LDA	A9	A5	B5	A1	B1	AD	BD	B9
CMP	C9	C5	D5	C1	D1	CD	DD	D9
SBC	E9	E5	F5	E1	F1	ED	FD	F9

Op Code ends in -1, -5, -9, or -D

Mde.	IMM	ZPg	Z.X	ABS	A.X
Byts	2	2	2	3	3
BIT		24		2C	
STY		84	94	8C	
LDY	A0	A4	B4	AC	BC
CPY	C0	C4		CC	
CPX	E0	E4		EC	

Op Code ends in -0, -4, or -C

Jim Butterfield

Branches -0			
BPL	10	BMI	30
BVC	50	BVS	70
BCC	90	BCS	B0
BNE	D0	BEQ	F0

Jumps		
Mde.	ABS	(IND)
JSR	20	
JMP	4C	6C

Mde.	IMM	ZPg	Z.X	Z.Y	ABS	A.X	A.Y
Byts	2	2	2	2	3	3	3
ASL		06	16		0E	1E	
ROL		26	36		2E	3E	
LSR		46	56		4E	5E	
ROR		66	76		6E	7E	
STX		86		96	8E		
LDX	A2	A6		B6	AE		BE
DEC		C6	D6		CE	DE	
INC		E6	F6		EE	FE	

Op Code ends in -2, -6, or -E

Single Byte Op Codes (* Accumulator Mode)

	0-	1-	2-	3-	4-	5-	6-	7-	8-	9-	A-	B-	C-	D-	E-	F-
-0	BRK				RTI		RTS									
-8	PHP	CLC	PLP	SEC	PHA	CLI	PLA	SEI	DEY	TYA	TAY	CLV	INY	CLD	INX	SED
-A	ASL*		ROL*		LSR*		ROR*		TXA	TXS	TAX	TSX	DEX		NOP	

6502 Extra Op-Codes

The table shows Op-Codes that are not generally recognized as part of the 650X Instruction Set.
Mnemonics and descriptions are from B. Grainger's article in IPUG (Jan 1981) and "Programming the PET/CBM" by Raeto Collin West

Instruction		Description	Abs	Abs,X	Abs,Y	Zer	Zer,X	Zer,Y	(Ind,X)	(Ind,Y)	Imm
ASO	(ASL, ORA)	ASL then ORA the result with the accumulator	0F	1F	1B	07	17		03	13	0B
RLA	(ROL, AND)	ROL then AND the result with the accumulator	2F	3F	3B	27	37		23	33	2B
LSE	(LSR, EOR)	LSR then EOR the result with the accumulator	4F	5F	5B	47	57		43	53	4B
RRA	(ROR, ADC)	ROR then ADC the result to the accumulator	6F	7F	7B	67	77		63	73	6B
AXS	(STX, STA)	Store the result of A AND X	8F			87		97	83		
LAX	(LDX, LDA)	LDA and LDX with the same data	AF		BF	A7	B7		A3	B3	
DCM	(DEC, CMP)	DEC memory then SBC the result from the accumulator	CF	DF	DB	C7	D7		C3	D3	
INS	(INC, SBC)	INC memory then SBC the result from the accumulator	EF	FF	FB	E7	F7		E3	F3	
ALR	(LSR, EOR)	AND the accumulator with data and LSR the result									4B
ARR	(ROR, ADC)	AND the accumulator with data and ROR the result									6B
XAA	(TXA,)	Store X AND data in the accumulator									8B
OAL	(TAX, LDA)	ORA the accumulator with #$EE, AND the result with data, then TAX									AB
SAX	(DEX, CMP)	SBC data from A AND X and store the result in X									CB
MKA	(AND, STA)	Store the result of .A AND #$04 in memory (Mask A bit 2)	9F								
MKX	(AND, STX)	Store the result of .X AND #$04 in memory (Mask X bit 2)	9E								
NOP		No operation	1A, 3A, 5A, 7A, DA, FA								
SKB		Skip next byte	80, 82, C2, E2, 04, 14, 34, 44, 54, 64, 74, D4, F4								
SKW		Skip next word (two bytes)	0C, 1C, 3C, 5C, 7C, DC, FC								

Hexadecimal Conversion Chart

Hex	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F	-00	-000
0-	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	0
1-	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	256	4096
2-	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	512	8192
3-	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	768	12288
4-	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	1024	16384
5-	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	1280	20480
6-	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	1536	24576
7-	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	1792	28672
8-	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	2048	32768
9-	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	2304	36864
A-	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	2560	40960
B-	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	2816	45056
C-	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	3072	49152
D-	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	3328	53248
E-	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	3584	57344
F-	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	3840	61440

Bit Values

Bit	Dec	Hex
0	1	$0001
1	2	0002
2	4	0004
3	8	0008
4	16	0010
5	32	0020
6	64	0040
7	128	0080
8	256	0100
9	512	0200
10	1024	0400
11	2048	0800
12	4096	1000
13	8192	2000
14	16384	4000
15	32768	8000

Instruction Set Summary

Instr	Addressing Mode	Assembler Format	Operation	Op Code Hex	Op Code Dec	Bytes	Clock Cycles	Status Register - P	Instr
ADC	Immediate	ADC #oper	.A + # + C → .A, C	69	105	2	2	N V D I Z C	**ADC**
	Zero Page	ADC addr	.A + [addr] + C → .A, C	65	101	2	3	✓ ✓ - - ✓ ✓	
	Zero Page, X	ADC addr, X	.A + [addr + .X] + C → .A, C	75	117	2	4		
	Absolute	ADC ADDR	.A + [ADDR] + C → .A, C	6D	109	3	4		
	Absolute, X	ADC ADDR, X	.A + [ADDR + .X] + C → .A, C	7D	125	3	4*		
	Absolute, Y	ADC ADDR, Y	.A + [ADDR + .Y] + C → .A, C	79	121	3	4*		
	(Indirect, X)	ADC (addr, X)	.A + [[addr + .X + 1, addr + .X]] + C → .A, C	61	97	2	6		
	(Indirect),Y	ADC (addr),Y	.A + [[addr + 1, addr] + .Y] + C → .A, C	71	113	2	5*		
AND	Immediate	AND #oper	.A ∩ # → .A	29	41	2	2	N V D I Z C	**AND**
	Zero Page	AND addr	.A ∩ [addr] → .A	25	37	2	3	✓ - - - ✓ -	
	Zero Page, X	AND addr, X	.A ∩ [addr + .X] → .A	35	53	2	4		
	Absolute	AND ADDR	.A ∩ [ADDR] → .A	2D	45	3	4		
	Absolute, X	AND ADDR, X	.A ∩ [ADDR + .X] → .A	3D	61	3	4*		
	Absolute, Y	AND ADDR, Y	.A ∩ [ADDR + .Y] → .A	39	57	3	4*		
	(Indirect, X)	AND (addr, X)	.A ∩ [[addr + .X + 1, addr + .X]] → .A	21	33	2	6		
	(Indirect),Y	AND (addr),Y	.A ∩ [[addr + 1, addr] + .Y] → .A	31	49	2	5*		
ASL	Accumulator	ASL A	.A (←) → .A ; 0 → bit 0, bit7 → C	0A	10	1	2	N V D I Z C	**ASL**
	Zero Page	ASL addr	[addr] (←) → [addr] "	06	6	2	5	✓ - - - ✓ ✓	
	Zero Page, X	ASL addr, X	[addr + .X] (←) → [addr + .X] "	16	22	2	6		
	Absolute	ASL ADDR	[ADDR] (←) → [ADDR] "	0E	14	3	6		
	Absolute, X	ASL ADDR, X	[ADDR + .X] (←) → [ADDR + .X] "	1E	30	3	7		
BCC	Relative	BCC oper	Branch on C = 0	90	144	2	2*	N V D I Z C	**BCC**
BCS	Relative	BCS oper	Branch on C = 1	B0	176	2	2*	- - - - - -	**BCS**
BEQ	Relative	BEQ oper	Branch on Z = 1	F0	240	2	2*	All Branches	**BEQ**
BNE	Relative	BNE oper	Branch on Z = 0	D0	208	2	2*	* - Add 1 if branch to	**BNE**
BMI	Relative	BMI oper	Branch on N = 1	30	48	2	2*	same page	**BMI**
BPL	Relative	BPL oper	Branch on N = 0	10	16	2	2*	* - Add 2 if branch to	**BPL**
BVS	Relative	BVS oper	Branch on V = 1	70	112	2	2*	diff page	**BVS**
BVC	Relative	BVC oper	Branch on V = 0	50	80	2	2*		**BVC**
BIT	Zero Page	BIT addr	.A ∩ [addr] ; bit7 → N, bit6 → V	24	36	2	3	N V D I Z C	**BIT**
	Absolute	BIT ADDR	.A ∩ [ADDR] "	2C	44	3	4	b_7 b_6 - - ✓ -	
BRK	Implied	BRK 1→B flag	PC + 2↓ P↓, [FFFE]→PCL, [FFFF]→PCH	00	0	1	7	- - - 1 - -	**BRK**
								N V D I Z C	
CLC	Implied	CLC	0→C	18	24	1	2	- - - - - 0	**CLC**
CLD	Implied	CLD	0→D	D8	216	1	2	- - 0 - - -	**CLD**
CLI	Implied	CLI	0→I	58	88	1	2	- - - 0 - -	**CLI**
CLV	Implied	CLV	0→V	B8	184	1	2	- 0 - - - -	**CLV**
CMP	Immediate	CMP #oper	.A – #	C9	201	2	2	N V D I Z C	**CMP**
	Zero Page	CMP addr	.A – [addr]	C5	197	2	3	✓ - - - ✓ ✓	
	Zero Page, X	CMP addr, X	.A – [addr + .X]	D5	213	2	4		
	Absolute	CMP ADDR	.A – [ADDR]	CD	205	3	4		
	Absolute, X	CMP ADDR, X	.A – [ADDR + .X]	DD	221	3	4*		
	Absolute, Y	CMP ADDR, Y	.A – [ADDR + .Y]	D9	217	3	4*		
	(Indirect, X)	CMP (addr, X)	.A – [[addr + .X + 1, addr + .X]]	C1	193	2	6		
	(Indirect),Y	CMP (addr),Y	.A – [[addr + 1, addr] + .Y]	D1	209	2	5*		
CPX	Immediate	CPX #oper	.X – #	E0	224	2	2	N V D I Z C	**CPX**
	Zero Page	CPX addr	.X – [addr]	E4	228	2	3	✓ - - - ✓ ✓	
	Absolute	CPX ADDR	.X – [ADDR]	EC	236	3	4		
CPY	Immediate	CPY #oper	.Y – #	C0	192	2	2	N V D I Z C	**CPY**
	Zero Page	CPY addr	.Y – [addr]	C4	196	2	3	✓ - - - ✓ ✓	
	Absolute	CPY ADDR	.Y – [ADDR]	CC	204	3	4		
DEC	Zero Page	DEC addr	[addr] – 1 → [addr]	C6	198	2	5	N V D I Z C	**DEC**
	Zero Page, X	DEC addr, X	[addr + .X] – 1 → [addr + .X]	D6	214	2	6	✓ - - - ✓ -	
	Absolute	DEC ADDR	[ADDR] – 1 → [ADDR]	CE	206	3	6		
	Absolute, X	DEC ADDR, X	[ADDR + .X] – 1 → [ADDR + .X]	DE	222	3	7		
DEX	Implied	DEX	.X – 1 → .X	CA	202	1	2	N V D I Z C	**DEX**
DEY	Implied	DEY	.Y – 1 → .Y	88	136	1	2	✓ - - - ✓ -	**DEY**
EOR	Immediate	EOR #oper	.A ⊎ # → .A	49	73	2	2	N V D I Z C	**EOR**
	Zero Page	EOR addr	.A ⊎ [addr] → .A	45	69	2	3	✓ - - - ✓ -	
	Zero Page, X	EOR addr, X	.A ⊎ [addr + .X] → .A	55	85	2	4		
	Absolute	EOR ADDR	.A ⊎ [ADDR] → .A	4D	77	3	4		
	Absolute, X	EOR ADDR, X	.A ⊎ [ADDR + .X] → .A	5D	93	3	4*		
	Absolute, Y	EOR ADDR, Y	.A ⊎ [ADDR + .Y] → .A	59	89	3	4*		
	(Indirect, X)	EOR (addr, X)	.A ⊎ [[addr + .X + 1, addr + .X]] → .A	41	65	2	6		
	(Indirect),Y	EOR (addr),Y	.A ⊎ [[addr + 1, addr] + .Y] → .A	51	81	2	5*		
INC	Zero Page	INC addr	[addr] + 1 → [addr]	E6	230	2	5	N V D I Z C	**INC**
	Zero Page, X	INC addr, X	[addr + .X] + 1 → [addr + .X]	F6	246	2	6	✓ - - - ✓ -	
	Absolute	INC ADDR	[ADDR] + 1 → [ADDR]	EE	238	3	6		
	Absolute, X	INC ADDR, X	[ADDR + .X] + 1 → [ADDR + .X]	FE	254	3	7		
INX	Implied	INX	.X + 1 → .X	E8	232	1	2	N V D I Z C	**INX**
INY	Implied	INY	.Y + 1 → .Y	C8	200	1	2	✓ - - - ✓ -	**INY**
JMP	Absolute	JMP ADDR	[PC + 1] → PCL, [PC + 2] → PCH	4C	76	3	3	N V D I Z C	**JMP**
	Indirect	JMP (ADDR)	[ADDR] → PCL, [ADDR + 1] → PCH	6C	108	3	5	- - - - - -	
JSR	Absolute	JSR ADDR	PC + 2↓, [PC + 1] → PCL, [PC + 2] → PCH	20	32	3	6		**JSR**

Instr	Addressing Mode	Assembler Format	Operation	Op Code Hex	Op Code Dec	Bytes	Clock Cycles	Status Register - P	Instr
LDA	Immediate	LDA #oper	# → .A	A9	169	2	2	N V D I Z C	**LDA**
	Zero Page	LDA addr	[addr] → .A	A5	165	2	3	✓ - - - ✓ -	
	Zero Page, X	LDA addr, X	[addr + .X] → .A	B5	181	2	4		
	Absolute	LDA ADDR	[ADDR] → .A	AD	173	3	4		
	Absolute, X	LDA ADDR, X	[ADDR + .X] → .A	BD	189	3	4*		
	Absolute, Y	LDA ADDR, Y	[ADDR + .Y] → .A	B9	185	3	4*		
	(Indirect, X)	LDA (addr, X)	[[addr + .X + 1, addr + .X]] → .A	A1	161	2	6		
	(Indirect),Y	LDA (addr),Y	[[addr + 1, addr] + .Y] → .A	B1	177	2	5*		
LDX	Immediate	LDX #oper	# → .X	A2	162	2	2	N V D I Z C	**LDX**
	Zero Page	LDX addr	[addr] → .X	A6	166	2	3	✓ - - - ✓ -	
	Zero Page, Y	LDX addr, Y	[addr + .Y] → .X	B6	182	2	4		
	Absolute	LDX ADDR	[ADDR] → .X	AE	174	3	4		
	Absolute, Y	LDX ADDR, Y	[ADDR + .Y] → .X	BE	190	3	4*		
LDY	Immediate	LDY #oper	# → .Y	A0	160	2	2	N V D I Z C	**LDY**
	Zero Page	LDY addr	[addr] → .Y	A4	164	2	3	✓ - - - ✓ -	
	Zero Page, X	LDY addr, X	[addr + .X] → .Y	B4	180	2	4		
	Absolute	LDY ADDR	[ADDR] → .Y	AC	172	3	4		
	Absolute, X	LDY ADDR, X	[ADDR + .X] → .Y	BC	188	3	4*		
LSR	Accumulator	LSR A	.A (→) → .A ; 0→bit7, bit0→C	4A	74	1	2	N V D I Z C	**LSR**
	Zero Page	LSR addr	[addr] (→) → [addr] ''	46	70	2	5	0 - - - ✓ ✓	
	Zero Page, X	LSR addr, Y	[addr + .X] (→) → [addr + .X] ''	56	86	2	6		
	Absolute	LSR ADDR	[ADDR] (→) → [ADDR] ''	4E	78	3	6		
	Absolute, X	LSR ADDR, X	[ADDR + .X] (→) → [ADDR + .X] ''	5E	94	3	7		
NOP	Implied	NOP	No OPeration	EA	234	1	2	- - - - - -	**NOP**
ORA	Immediate	ORA #oper	.A ∪ # → .A	09	9	2	2	N V D I Z C	**ORA**
	Zero Page	ORA addr	.A ∪ [addr] → .A	05	5	2	3	✓ - - - ✓ -	
	Zero Page, X	ORA addr, X	.A ∪ [addr + .X] → .A	15	21	2	4		
	Absolute	ORA ADDR	.A ∪ [ADDR] → .A	0D	13	3	4		
	Absolute, X	ORA ADDR, X	.A ∪ [ADDR + .X] → .A	1D	29	3	4*		
	Absolute, Y	ORA ADDR, Y	.A ∪ [ADDR + .Y] → .A	19	25	3	4*		
	(Indirect, X)	ORA (addr, X)	.A ∪ [[addr + .X + 1, addr + .X]] → .A	01	1	2	6		
	(Indirect),Y	ORA (addr),Y	.A ∪ [[addr + 1, addr] + .Y] → .A	11	17	2	5*		
PHA	Implied	PHA	.A ↓, SP - 1 → SP	48	72	1	3	N V D I Z C	**PHA**
PLA	Implied	PLA	.A ↑, SP + 1 → SP	68	104	1	4	- - - - - -	**PLA**
PHP	Implied	PHP	.P ↓, SP - 1 → SP	08	8	1	3	All Push/Pulls xcpt PLP	**PHP**
PLP	Implied	PLP	.P ↑, SP + 1 → SP	28	40	1	4	from stack	**PLP**
ROL	Accumulator	ROL A	.A (←) → .A ; C→bit0, bit7→C	2A	42	1	2	N V D I Z C	**ROL**
	Zero Page	ROL addr	[addr] (←) → [addr] ''	26	38	2	5	✓ - - - ✓ ✓	
	Zero Page, X	ROL addr, X	[addr + .X] (←) → [addr + .X] ''	36	54	2	6		
	Absolute	ROL ADDR	[ADDR] (←) → [ADDR] ''	2E	46	3	6		
	Absolute, X	ROL ADDR, X	[ADDR + .X] (←) → [ADDR + .X] ''	3E	62	3	7		
ROR	Accumulator	ROR A	.A (→) → .A ; C→bit7, bit0→C	6A	106	1	2	N V D I Z C	**ROR**
	Zero Page	ROR addr	[addr] (→) → [addr] ''	66	102	2	5	✓ - - - ✓ ✓	
	Zero Page, X	ROR addr, Y	[addr + .X] (→) → [addr + .X] ''	76	118	2	6		
	Absolute	ROR ADDR	[ADDR] (→) → [ADDR] ''	6E	110	3	6		
	Absolute, X	ROR ADDR, X	[ADDR + .X] (→) → [ADDR + .X] ''	7E	126	3	7		
RTI	Implied	RTI	P ↑, PC ↑, SP + 3 → SP, PC + 1 → PC	40	64	1	6	from stack	**RTI**
RTS	Implied	RTS	PC ↑, SP + 2 → SP, PC + 1 → PC	60	96	1	6	- - - - - -	**RTS**
SBC	Immediate	SBC #oper	.A - # - C̄ → .A, C C̄ = Borrow	E9	233	2	2	N V D I Z C	**SBC**
	Zero Page	SBC addr	.A - [addr] - C̄ → .A, C	E5	229	2	3	✓ ✓ - - ✓ ✓	
	Zero Page, X	SBC addr, X	.A - [addr + .X] - C̄ → .A, C	F5	245	2	4		
	Absolute	SBC ADDR	.A - [ADDR] - C̄ → .A, C	ED	237	3	4		
	Absolute, X	SBC ADDR, X	.A - [ADDR + .X] - C̄ → .A, C	FD	253	3	4*		
	Absolute, Y	SBC ADDR, Y	.A - [ADDR + .Y] - C̄ → .A, C	F9	249	3	4*		
	(Indirect, X)	SBC (addr, X)	.A - [[addr + .X + 1, addr + .X]] - C̄ → .A, C	E1	225	2	6		
	(Indirect),Y	SBC (addr),Y	.A - [[addr + 1, addr] + .Y] - C̄ → .A, C	F1	241	2	5*		
								N V D I Z C	
SEC	Implied	SEC	1→C	38	56	1	2	- - - - - 1	**SEC**
SED	Implied	SED	1→D	F8	248	1	2	- - 1 - - -	**SED**
SEI	Implied	SEI	1→I	78	120	1	2	- - - 1 - -	**SEI**
STA	Zero Page	STA addr	.A → [addr]	85	133	2	3	N V D I Z C	**STA**
	Zero Page, X	STA addr, X	.A → [addr + .X]	95	149	2	4	- - - - - -	
	Absolute	STA ADDR	.A → [ADDR]	8D	141	3	4		
	Absolute, X	STA ADDR, X	.A → [ADDR + .X]	9D	157	3	5		
	Absolute, Y	STA ADDR, Y	.A → [ADDR + .Y]	99	153	3	5		
	(Indirect, X)	STA (addr, X)	.A → [[addr + .X + 1, addr + .X]]	81	129	2	6		
	(Indirect),Y	STA (addr),Y	.A → [[addr + 1, addr] + .Y]	91	145	2	6		
STX	Zero Page	STX addr	.X → [addr]	86	134	2	3	N V D I Z C	**STX**
	Zero Page, Y	STX addr, Y	.X → [addr + .Y]	96	150	2	4	- - - - - -	
	Absolute	STX ADDR	.X → [ADDR]	8E	142	3	4		
STY	Zero Page	STY addr	.Y → [addr]	84	132	2	3	N V D I Z C	**STY**
	Zero Page, X	STY addr, X	.Y → [addr + .X]	94	148	2	4	- - - - - -	
	Absolute	STY ADDR	.Y → [ADDR]	8C	140	3	4		
TAX	Implied	TAX	.A → .X	AA	170	1	2	N V D I Z C	**TAX**
TXA	Implied	TXA	.X → .A	8A	138	1	2	✓ - - - ✓ -	**TXA**
TAY	Implied	TAY	.A → .Y	A8	168	1	2		**TAY**
TYA	Implied	TYA	.Y → .A	98	152	1	2	All Transfers xcpt TXS	**TYA**
TSX	Implied	TSX	SP → .X	BA	186	1	2		**TSX**
TXS	Implied	TXS	.X → SP	9A	154	1	2	- - - - - -	**TXS**

MCS65XX Microprocessor Instruction Set

Mnemonic	Definition
ADC	Add memory to accumulator with carry.
AND	AND memory with accumulator.
ASL	Shift left one bit (memory or accumulator).
BCC	Branch on carry clear.
BCS	Branch on carry set.
BEQ	Branch on result zero.
BIT	Test bits in memory with accumulator.
BMI	Branch on result minus.
BNE	Branch on result not zero.
BPL	Branch on result plus.
BRK	Force break.
BVC	Branch on overflow clear.
BVS	Branch on overflow set.
CLC	Clear carry flag.
CLD	Clear decimal mode.
CLI	Clear interrupt disable bit.
CLV	Clear overflow flag.
CMP	Compare memory and accumulator.
CPX	Compare memory and index 'X'.
CPY	Compare memory and index 'Y'.
DEC	Decrement memory by one.
DEX	Decrement index 'X' by one.
DEY	Decrement index 'Y' by one.
EOR	Exclusive-OR memory with accumulator.
INC	Increment memory by one.
INX	Increment index 'X' by one.
INY	Increment index 'Y' by one.
JMP	Jump to new location.
JSR	Jump to new location saving return address.
LDA	Load acculmulator with memory.
LDX	Load index 'X' with memory.
LDY	Load index 'Y' with memory.
LSR	Shift right one bit (memory or accumulator).
NOP	No operation.
ORA	OR memory with accumulator.
PHA	Push accumulator on stack.
PHP	Push processor status on stack.
PLA	Pull accumulator from stack.
PLP	Pull processor status from stack.
ROL	Rotate one bit left (memory or accumulator).
ROR	Rotate one bit right (memory or accumulator).
RTI	Return from interrupt.
RTS	Return from subroutine.
SBC	Subtract memory from accumulator with borrow.
SEC	Set carry flag.
SED	Set decimal mode.
SEI	Set interrupt disable status.
STA	Store accumulator in memory.
STX	Store index 'X' in memory.
STY	Store index 'Y' in memory.
TAX	Transfer accumulator to index 'X'.
TAY	Transfer accumulator to index 'Y'.
TSX	Transfer stack pointer to index 'X'.
TXA	Transfer index 'X' to accumulator.
TXS	Transfer index 'X' to stack pointer.
TYA	Transfer index 'Y' to accumulator.

Addressing Modes

Accumulator Addressing - This form of addressing is represented with a one byte instruction, implying an operation on the accumulator.

Immediate Addressing - In immediate addressing, the operand is contained in the second byte of the instruction, with no further memory addressing required.

Absolute Addressing - In absolute addressing, the second byte of the instruction specifies the eight low order bits of the effective address while the third byte specifies the eight high order bits. Thus, the absolute addressing mode allows access to the entire 65k bytes of addressable memory.

Zero Page Addressing - The zero page instructions allow for shorter code and execution times by only fetching the second byte of the instructions and assuming a zero high address byte. Careful use of the zero page can result in significant increase in code efficiency.

Indexed Zero Page Addressing - (X, Y Indexing) - This form of addressing is used in conjunction with the index register and is referred to as "Zero Page, X" or "Zero Page, Y". The effective address is calculated by adding the second byte to the contents of the index register. Since this is a form of "Zero Page" addressing, the content of the second byte references a location in page zero. Additionally due to the "Zero Page" addressing nature of this mode, no carry is added to the high order 8 bits of memory and crossing of page boundaries does not occur.

Indexed Absolute Addressing - (X, Y Indexing) - This form of addressing is used in conjunction with X and Y index register and is referred to as Absolute, X", and "Absolute, Y". The effective address is formed by adding the contents of X or Y to the address contained in the second and third bytes on the instruction. This mode allows the index register to contain the index or count value and the instruction to contain the base address. This type of indexing allows any location referencing and the index to modify multiple fields resulting in reduced coding and execution time.

Implied Addressing - In the implied addressing mode, the address containing the operand is implicitly stated in the operation code of the instruction.

Relative Addressing - Relative addressing is used only with branch instructions and establishes a destination for the conditional branch. The second byte of the instruction becomes the operand which is an "offset" added to the contents of the lower eight bits of the program counter when the counter is set at the next instruction. The range of the offset is -128 to +127 bytes from the next instruction.

Indexed Indirect Addressing - In indexed indirect addressing (referred to as (Indirect, X)), the second byte of the instruction is added to the contents of the X index register, discarding the carry. The result of the addition points to a memory location on page zero whose contents is the low order eights bits of the effective address. The next memory location in page zero contains the high order eight bits of the effective address. Both memory locations specifying the high and low order bytes of the effective address must be in page zero.

Indirect Indexed Addressing - In indirect indexed addressing (referred to as (Indirect), Y), the second byte of the instruction points to a memory location in page zero. The contents of this memory location is added to the contents of the Y register, the result being the low order eight bits of the effective address. The carry from this addition is added to the contents of the next page zero memory location, the result being the high order eight bits of the effective address.

Absolute Indirect - The second byte of the instruction contains the low order eight bits of a memory location. The high order eight bits of that memory location is contained in the third byte of the instruction. The contents of the fully specified memory location is the low order byte of the effective address which is loaded into the sixteen bits of the program counter.

User Callable ROM Subroutines

Some I/O routines require extra memory set up. See the appropriate Memory Map. Address pairs within parenthesis are for Basic 2.0/4.0 users. (Direct call) indicates no required set up.

#	Entry Point For: 2.0		4.0		VIC 20		C64		Operation	Registers In .A	.X	.Y	Registers Out .A	.X	.Y
1	C2D8	49880	B350	45904	C3BB	50107	A3BB	41915	Open Up Space In BASIC Text New:	AryTop Lo		AryTop Hi	Unaltered		
2	C328	49960	B3A0	45984	C408	50184	A408	41992	Check Available Memory (called by 1)	(same as above) Start address of move in $5F, 60 ($5C, 5D)					
3	C355	50005	B3CD	46029	C435	50229	A435	42037	?OUT OF MEMORY	(direct call)					
4	C357	50007	BC3F	48191	C437	50231	A437	42039	Send BASIC Error Message	Error #					
5	C389	50057	B3FF	46079	C474	50292	A474	42100	Warm start, BASIC	(direct call)					
6	C399	49960	B40D	46093	C48A	50314	A48A	42122	Main CHRGET entry	(direct call) $7A = #$FF, $7B = #$01 ($77, 78) ;01FF = Basic Inbuf−1					
7	C3AB	50091	B41F	46111	C49C	50220	A49C	42028	Crunch tokens, insert line		Inbuf len.				
8	C439	50233	B4AD	46253	C52A	50474	A52A	42282	Fix chaining, CLR, & READY.	(direct call)					
9	C442	50242	B4B6	46262	C533	50483	A533	42291	Fix chaining	(direct call)					
10	C46F	50287	B4E2	46306	C560	50528	A560	42336	Receive line from keyboard	(direct call) $7A = #$FF, $7B = #$01 ($77, 78) ;01FF = Basic Inbuf−1					
11	C495	50213	B4FB	46331	C579	50553	A579	42361	Crunch tokens (called by 7)	.X = Inbuf Len. ($0200,X) = #$00					
12	C52C	50476	B5A3	46499	C613	50707	A613	42515	Find line in BASIC	StrtBAS Lo	StrtBAS Hi				
13	C55D	50525	B5D4	46548	C642	50754	A642	42562	Do NEW	(direct call)					
14	C572	50546	B5E9	46569	C659	50777	A659	42585	Reset BASIC and do CLR	(direct call)					
15	C575	50549	B5EC	46572	C65E	50782	A65E	42590	Do CLR	(direct call)					
16	C597	50583	B612	46610	n/a	n/a	n/a	n/a	Purge stack of all Returns & Nexts (POP)	(direct call)					
17	C5A7	50599	B622	46626	C68E	50830	A68E	42638	Reset Chrget to Start of BASIC	(direct call)			StrtBAS Hi		
18	C6C4	50884	B74A	46922	C857	51287	A857	43095	Continue BASIC execution [CONT]	CurLin Lo		CurLin Hi			
19	C873	51315	B8F6	47350	C96B	49771	A96B	41579	Get fixed-pt number from BASIC text	Address of text in Chrget ptr; $7A, 7B ($77, 78)					
20	C9DE	49886	BADB	47835	CAD3	51923	AAD3	43731	Send RETURN, LF if in screen mode	(direct call)			LF (#$0A)		
21	C9E2	49890	BADF	47839	CAD7	51927	AAD7	43735	Send RETURN, LINEFEED	(direct call)			LF (#$0A)		
22	CA1C	51740	BB1D	47901	CB1E	51998	AB1E	43806	Print string from A, Y	Addr Lo		Addr Hi			
23	CA22	51746	BB23	47907	CB24	52004	AB24	43812	Print pre-computed string	Length	Addr in $22,23 ($1F,20)				
24	CA43	51779	BB44	47940	CB45	52037	AB45	43845	Print '?'	(direct call)					
25	CA45	51781	BB46	47942	CB47	52039	AB47	43847	Print char (output .A to device)	Char			Char		
26	CC9F	52383	BD98	48536	CD9E	52638	AD9E	44446	Evaluate Expression Result: string $0D = #$FF ($07) numeric $0D = #$00 ($07)	Address of Expression In Chrget Pointer			Addr Lo result in Acc#1		Addr Hi
27	CDF8	52728	BEF5	48885	CEFF	52991	AEFD	44797	Check for comma	(direct call)			Char		
28	CDF7	52727	BEF2	48882	CEFA	52986	AEFA	44794	Check for '('	(direct call)			Char		
29	CDF4	52724	BEEF	48879	CEF7	52983	AEF7	44791	Check for ')'	(direct call)			Char		
30	CE03	52739	BF00	48896	CF08	53000	AF08	44808	Send 'SYNTAX ERROR'	(direct call)					
31	CFC9	53193	C187	49543	D0E7	53479	B0E7	45287	Find fl-pt variable, given name				VarAddr Lo		VarAddr Hi
32	D069	53353	C2B9	49849	D185	53637	B185	45445	Bump Variable Addr by 2 (called by 31)	Name in $45, 46 ($42, 43)			VarAddr Lo		VarAddr Hi
33	D09A	53290	C2EA	49898	D1BF	53695	B1BF	45503	Float to Fixed conversion in Acc#1	(direct call)					
34	D26D	53869	C4BC	50364	D391	54049	B391	45857	Fixed to Float conversion in Acc#1	(direct call)					
35	D67B	54907	C8D7	51415	D79E	55086	B79E	46894	Get Acc#1 least significant byte to X register					Data	
36	D68F	54927	C8EB	51435	D7B5	55221	B7B5	47029	Evaluate string [VAL]	Address = (Chrget Ptr.)			Fl. Pt. result in Acc#1		
37	D69D	54931	C8EF	51439	D7B9	55225	B7B9	47033	Evaluate string from X, Y (above + 4)		Addr Lo	Addr Hi	Fl. Pt. result in Acc#1		
38	D6C6	54982	C921	49697	D7EB	55275	B7EB	47083	Get two params for POKE, WAIT	Address = (Chrget Ptr.)			.X = Pram2, Pram1 in Acc#1 (fxd pt)		
39	D773	55155	C99D	49709	D867	55399	B867	47207	Add (from memory)	Addr Lo		Addr Hi	Fl. Pt. result in Acc#1		
40	D934	53812	CB5E	52062	DA28	55848	BA28	47656	Multiply from memory location	Addr Lo		Addr Hi	Fl. Pt. result in Acc#1		
41	D9EE	53998	CC18	52248	DAE2	56034	BAE2	47842	Multiply Acc#1 by ten				(result in Acc#1)		
42	DAAE	55982	CCD8	52440	DBA2	56226	BBA2	48034	Unpack memory variable to Acc#1	Addr Lo		Addr Hi			
43	DAE3	56035	CD0D	52493	DBD7	56279	BBD7	48087	Copy Acc#1 to (X,Y) Location	Addr Lo	Addr Hi				
44	DB08	56072	CD32	52530	DBFC	56316	BBFC	48124	Move Acc#2 to Acc#1	(direct call)					
45	DB18	56088	CD42	52546	DC0C	56332	BC0C	48140	Move Rounded Acc#1 to Acc#2	(direct call)					
46	DB1D	56093	CD45	52549	DC0F	56335	BC0F	48143	Move Un-Rounded Acc#1 to Acc#2	(direct call)					
47	DB27	56103	CD51	52561	DC1B	56347	BC1B	48155	Round Acc.#1	(direct call)					
48	DCD9	56537	CF83	53123	DDCD	56781	BDCD	48589	Print fixed-point value	Value Hi	Value Lo				
49	DCE3	56547	CF8D	53133	DDD7	56791	BDD7	48599	Print floating-point value in Acc#1	(direct call)					
50	DCE9	56553	CF93	53027	DDDD	56797	BDDD	48605	Cnvert num to strng at $0100 (calld by 48)	#$00		#$01			
51	FD11	64785	D472	54386	n/a	n/a	n/a	n/a	Entry to M.L.M.	(direct call)					
52	E3D8	58328	E202	57858	E742	59202	E716	59158	Print a character	Char					
53	F156	61782	F185	61829	F1E6	61926	F12F	61743	Print system message			Offset			
54	F0B6	61622	F0D2	61650	EE14	60948	ED09	60681	Send 'talk' to IEEE/Serial	Dev #					
55	F0BA	61626	F0D5	61653	EE17	60951	ED0C	60684	Send 'listen' to IEEE/Serial	Dev #					
56	F128	61736	F143	61763	FF93	65427	FF93	65427	Send secondary address	SA OR $60					
57	F16F	61807	F19E	61742	EEE4	61156	ED40	60736	Send char to IEEE/Serial	Char					
58	F17F	61823	F1AE	61870	EEF6	61174	EDEF	60911	Send 'untalk'	(direct call)					
59	F183	61827	F1B9	61881	EF04	61188	EDFE	60926	Send 'unlisten'	(direct call)					

BASIC 4.0 / 2.0 Kernal Routines

CBM Label	Address Hex	Address Dec	Operation	Registers In .A	Registers In .X	Registers In .Y	Registers Out .A	Registers Out .X	Registers Out .Y
CHKIN	FFC6	65478	Open channel for input		LF#		alt.		
CHKOUT	FFC9	65481	Open channel for output		LF#		alt.		
CHRIN	FFCF	65487	Input character from channel				data	alt.	
CHROUT	FFD2	65490	Output character to channel	data					
CLALL	FFE7	65511	Close all channels and files				alt.	alt.	
CLOSE	FFC3	65475	Close a specified logical file	LF#			alt.	alt.	alt.
CLRCHN	FFCC	65484	Restore default I/O devices				alt.	alt.	
CSYS	FFDE	65502	SYS vector		addr lo	addr hi	alt.	alt.	alt.
CVERF	FFDB	65499	Verify ram from a device		start lo	start hi		end lo + 1	end hi
GETIN	FFE4	65508	Get character from current input device				data	alt.	alt.
LOAD	FFD5	65493	Load ram from a device		start lo	start hi		end lo + 1	end hi
OPEN	FFC0	65472	Open a logical file				alt.	alt.	alt.
SAVE	FFD8	65496	Save 'ram' to device, from $28,29 to .X,.Y	#<txttab (=#$28)	end lo	end hi		end lo + 1	end hi
STOP	FFE1	65505	Scan stop key depressed	yes: .Z = 1, no .A = last row kybd scan					
UDTIM	FFEA	65514	Increment real time clock				alt.	alt.	

alt. = altered

VIC 20 And Commodore 64 Kernal Routines

CBM Label	Address Hex	Address Dec	Operation	Registers In .A	Registers In .X	Registers In .Y	Registers Out .A	Registers Out .X	Registers Out .Y
ACPTR	FFA5	65445	Input byte from Serial Port				data	alt.	
CHKIN	FFC6	65478	Open channel for input		LF#		alt.		
CHKOUT	FFC9	65481	Open channel for output		LF#		alt.		
CHRIN	FFCF	65487	Input character from channel				data	alt.	
CHROUT	FFD2	65490	Output character to channel	data					
CIOUT	FFA8	65448	Output byte to serial port	data					
CINT	FF81	65409	Initialize screen editor				alt.	alt.	alt.
CLALL	FFE7	65511	Close all channels and files				alt.	alt.	
CLOSE	FFC3	65475	Close a specified logical file	LF#			alt.	alt.	alt.
CLRCHN	FFCC	65484	Restore default I/O devices				alt.	alt.	
GETIN	FFE4	65508	Get character from current input device				data	alt.	alt.
IOBASE	FFF3	65523	Returns base address of I/O devices					addr lo	addr hi
IOINIT	FF84	65412	Initialize Input/Output				alt.	alt.	alt.
LISTEN	FFB1	65457	Command devices on the serial bus to listen	DEV#					
LOAD	FFD5	65493	Load (.A = 0) or Verify (.A = 1) 'ram' from a device		start lo	start hi		end lo + 1	end hi
MEMBOT	FF9C	65436	Read (.C = 1) or Set (.C = 0) the bottom of memory	.C = 0:	bot lo	bot hi	.C = 1:	bot lo	bot hi
MEMTOP	FF99	65433	Read (.C = 1) or Set (.C = 0) the top of memory	.C = 0:	top lo	top hi	.C = 1:	top lo	top hi
OPEN	FFC0	65472	Open a logical file				alt.	alt.	alt.
PLOT	FFF0	65520	Read (.C = 1) or Set (.C = 0) x, y cursor position		row	col		row	col
RAMTAS	FF87	65415	Init. ram, allocate tape buff, set screen $0400				alt.	alt.	alt.
RDTIM	FFDE	65502	Read real time clock				msb	msb2	lsb
READST	FFB7	65463	Read I/O status word				ST		
RESTOR	FF8A	65418	Restore default I/O vectors				alt.	alt.	alt.
SAVE	FFD8	65496	Save 'ram' to device, from $2B,2C to .X,.Y	#<txttab (=#$2B)	end lo	end hi		end lo + 1	end hi
SCNKEY	FF9F	65439	Scan keyboard				alt.	alt.	alt.
SCREEN	FFED	65517	Return screen size in rows & columns					#rows	#cols
SECOND	FF93	65427	Send secondary address after 'listen'	SA OR $60					
SETLFS	FFBA	65466	Set logical, first, and second addresses	LF#	DEV#	SA			
SETMSG	FF90	65424	Enable/Disable 'Kernal' messages	.A val: $40 control msgs on, $80 error msgs on, $00 off					
SETNAM	FFBD	65469	Set file name	len	addr lo	addr hi			
SETTIM	FFDB	65499	Set real time clock	msb	msb2	lsb			
SETTMO	FFA2	65442	Set (.A<#128) Reset (.A>#127) Serial/IEEE timeout						
STOP	FFE1	65505	Scan stop key depressed	yes: .Z = 1, no .A = last row kybd scan					
TALK	FFB4	65460	Command serial bus device to 'talk'	DEV#					
TKSA	FF96	65430	Send secondary address after 'talk'	SA					
UDTIM	FFEA	65514	Increment real time clock				alt.	alt.	
UNLSN	FFAE	65454	Command serial bus to 'unlisten'				alt.		
UNTLK	FFAB	65451	Command serial bus to 'untalk'				alt.		
VECTOR	FF8D	65421	Store (.C = 1) or Restore (.C = 0) ram vectors	.C = 1:	tabl lo	tabl hi	.C = 0:	tabl lo	tabl hi

alt. = altered

#	Entry Point For: 2.0		4.0		VIC 20		C64		Operation	Registers In .A	.X	.Y	Registers Out .A	.X	.Y
60	F18C	61836	F1C0	61888	EF19	61209	EE13	60947	Input from IEEE/Serial				Data		
61	F2A9	62121	F2DD	62173	F34A	62282	F291	61985	Close logical file (kernal rtn)	LF #					
62	F301	62209	F335	62261	F770	63344	F6ED	63213	Check for STOP key					Z flag = 1 if pressed	
63	F322	62242	F356	62294	F542	62786	F49E	62510	LOAD subroutine	#$00	Start Lo	Start Hi			
64	F40A	62474	F449	62537	F647	63047	F5AF	62895	Print SEARCHING...		(direct call)				
65	F41D	62493	F45C	62556	F659	63065	F5C1	62913	Print file name		(direct call)				
66	F494	62500	F4D3	62675	F867	63591	F7EA	63466	Find specific tape header block	Len	Pointer to string in $BB, BC (same for 2/4.0)				
67	F5A6	62886	F5E5	62949	F7AF	63407	F72D	63277	Find any tape header block		(direct call)				
68	F812	63506	F857	63575	F894	63524	F817	63511	Press PLAY...; wait		(direct call)				
69	F855	63573	F89A	63530	F8C0	63680	F841	63553	Read tape to buffer		(direct call)				
70	F85E	63582	F8A3	63651	F8C6	63686	F847	63559	Read tape		(direct call)				
71	F886	63622	F8CB	63691	F8E3	63715	F864	63588	Write tape from buffer						
72	F88E	63630	F8D3	63699	F8E8	63720	F869	63593	Write tape, leader length in A	Ldr Len.					
73	FB76	64374	FBBB	64443	FCF6	64758	FB8E	64398	Reset tape I/O		(direct call)				
74	FC9B	64555	FCE0	64736	FCF9	64761	FCBD	64701	Set interrupt vector		(direct call)				
75	FCD1	64721	FD16	64790	FD22	64802	FCE2	64738	Power On Reset		(direct call)				

BASIC Keyword Tokens and Entry Points

Keyword	Token Hex	Dec	ROM Entry Point BASIC 2.0		BASIC 4.0		VIC 20		C64	
ABS	B6	182	DB64	56164	CD8E	52622	DC58	59408	BC58	48216
AND	AF	175	CECB	52939	C089	49289	CFE9	53225	AFE9	45033
APPEND**	D4	212			FFAB	65451				
ASC	C6	198	D665	54885	C8C1	51393	D78B	55179	B78B	46987
ATN	C1	193	E08C	57484	D32C	54060	E30B	58123	E30E	58126
BACKUP**	D2	210			FFA5	65445				
CATALOG**	DA	215			FFB4	65460				
CHR	C7	199	D5C6	54726	C822	51234	D6E6	55020	B6EC	46828
CLOSE*	A0	160	FFC3	65475	FFC3	65475	FFC3	65475	FFC3	65475
CLR	9C	156	C577	50551	B5EE	46574	C65E	50782	A65E	42590
CMD	9D	157	C991	51601	BA8E	47758	CA86	51846	AA86	43654
COLLECT**	D1	209			FFA2	65442				
CONCAT**	CC	204			FF93	65427				
CONT	9A	154	C76B	51051	B7EE	47086	C857	51287	A857	43095
COPY**	D3	211			FFA8	65448				
COS	BE	190	DFD8	57304	D282	53890	E261	57953	E264	57956
DATA	83	131	C800	51200	B883	47235	C858	51448	A8F8	43256
DCLOSE**	CE	206			FF99	65433				
DEF	96	150	D28D	53901	C4DC	50396	D3B3	54195	B3B3	46003
DIM	86	134	CF63	53091	C121	49441	D081	53377	B081	45185
DIRECTORY**	DA	218			FFB4	65460				
DLOAD**	CD	205			FF96	65430				
DSAVE**	D5	213			FFAE	65454				
END	80	128	C741	51009	B7C8	47048	C831	51249	A831	43057
EXP	BD	189	DEDA	57050	D184	53636	DFED	57325	BFED	49133
FN	A5	165	D2CE	53966	C51D	50461	D3F4	54260	B3F4	46068
FOR	81	129	C658	50776	B6DE	46814	C742	51010	A742	42818
FRE	B8	184	D259	53849	C4A8	50344	D37D	54141	B37D	45949
GET*	A1	161	FFE4	65508	FFE4	65508	FFE4	65508	FFE4	65508
GOSUB	8D	141	C790	51088	B813	47123	C883	51331	A883	43139
GOTO	89	137	C7AD	51117	B830	47152	C8A0	51360	A8A0	43168
HEADER**	D0	208			FF9F	65439				
IF	8B	139	C830	51248	B8B3	47283	C928	51496	A928	43304
INPUT*	85	133	FFCF	65487	FFCF	65487	FFCF	65487	FFCF	65487
INPUT#	84	132	CAA7	51879	BBA4	48036	CBA5	52133	ABA5	43941
INT	B5	181	DBD8	56280	CE02	52738	DCCC	56524	BCCC	48332
LEFT	C8	200	D5DA	54746	C836	51254	D700	55040	B700	46848
LEN	C3	195	D656	54870	C8B2	51378	D77C	55164	B77C	46972
LET	88	136	C8AD	51373	B930	47408	C9A5	51621	A9A5	43429

Keyword	Token Hex	Dec	BASIC 2.0		BASIC 4.0		VIC 20		C64	
LIST	9B	155	C5B5	50613	B630	46640	C69C	50844	A69C	42652
LOAD*	93	147	FFD5	65493	FFD5	65493	FFD5	65493	FFD5	65493
LOG	BC	188	D8F6	55542	CB20	52000	D9EA	55786	B9EA	47594
MID	CA	202	D611	54801	C86D	51309	D737	55095	B737	46903
NEW	A2	162	C55B	50523	B5D2	46546	C642	50754	A642	42562
NEXT	82	130	CC20	52256	BD19	48409	CD1E	52510	AD1E	44318
NOT	A8	168	CDCF	52687	BECC	48844	CED4	52948	AED4	44756
ON	91	145	C853	51283	B8D6	47318	C94B	51531	A94B	43339
OPEN*	9F	159	FFC0	65472	FFC0	65472	FFC0	65472	FFC0	65472
OR	B0	176	CEC8	52936	C086	49286	CFE6	53222	AFE6	45030
PEEK	C2	194	D6E8	55016	C943	51523	D80D	55309	B80D	47117
POKE	97	151	D707	55047	C95A	51546	D824	55332	B824	47140
POS	B9	185	D27A	53882	C4C9	50377	D39E	54174	B39E	45982
PRINT*	99	153	FFD2	65490	FFD2	65490	FFD2	65490	FFD2	65490
PRINT#	98	152	C98B	51595	BA88	47752	CA80	51840	AA80	43648
READ	87	135	CB07	51975	BC02	48130	CC06	52230	AC06	44038
RECORD**	CF	207			FF9C	65436				
REM	8F	143	C843	51267	B8C6	47302	C93B	51515	A93B	43323
RENAME**	D8	216			FFB7	65463				
RESTORE	8C	140	C730	50992	B7B7	47031	C81D	51229	A81D	43037
RETURN	8E	142	C7DA	51162	B85D	47197	C8D2	51410	A8D2	43218
RIGHT	C9	201	D606	54790	C862	51298	D72C	55084	B72C	46892
RND	BB	187	DF7F	57215	D229	53801	E094	57492	E097	57495
RUN	8A	138	C785	51077	B808	47112	C871	51313	A871	43121
SAVE*	94	148	FFD8	65496	FFD8	65496	FFD8	65496	FFD8	65496
SCRATCH**	D9	217			FFBA	65466				
SGN	B4	180	BD45	56133	CD6F	52591	DC39	56377	BC39	48185
SIN	BF	191	DFDF	57311	D289	53897	E268	57960	E26B	57963
SPC(	A6	166	C9FC	51708	BAFD	47869	CAF8	51960	AAF8	43768
SQR	BA	186	DE5E	56926	D108	53512	DF71	57201	BF71	49009
STEP	A9	169	C6AB	50859	B731	46897	C795	51093	A795	42901
STOP	90	144	C73F	51007	B7C6	47046	C82F	51247	A82F	43055
STR	C4	196	D33F	54079	C58E	50574	D465	54373	B465	46181
SYS*	9E	158	F684	63108	F6C3	63171	E127	57639	E12A	57642
TAB(	A3	163	C9FC	51708	BAFD	47869	CAF8	51960	AAF8	43768
TAN	C0	192	E028	57384	D2D2	53970	E2B1	58033	E2B4	58036
USR	B7	183	PET/CBM JMP $0000. VIC/64 JMP($0311) ;USR Jump Vector							
VAL	C5	197	D687	54919	C8E3	51427	D7AD	55213	B7AD	47021
VERIFY*	95	149	FFDB	65499	FFDB	65499	FFDB	65499	FFDB	65499
WAIT	92	146	D710	55056	C963	51555	D82D	55341	B82D	47149

* Kernal Routine / ** BASIC 4.0 Kernal Routine

SuperChart: BASIC 2.0 / 4.0

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
0	00		@	end-line	BRK	0
1	01		A		ORA(I,X)	1
2	02		B			2
3	03	stop	C			3
4	04		D			4
5	05		E		ORA Z	5
6	06		F		ASL Z	6
7	07	bell	G			7
8	08		H		PHP	8
9	09	tab	I		ORA #	9
10	0A		J		ASL A	10
11	0B		K			11
12	0C		L			12
13	0D	car ret	M		ORA	13
14	0E	text	N		ASL	14
15	0F	top left	O			15
16	10		P		BPL	16
17	11	cur down	Q		ORA(I),Y	17
18	12	reverse	R			18
19	13	cur home	S			19
20	14	delete	T			20
21	15	del line	U		ORA Z,X	21
22	16	ers start	V		ASL Z,X	22
23	17		W			23
24	18		X		CLC	24
25	19	scroll dn	Y		ORA Y	25
26	1A		Z			26
27	1B	escape	[			27
28	1C		\			28
29	1D	cur right	]		ORA X	29
30	1E		↑		ASL X	30
31	1F		←			31
32	20	space	space	space	JSR	32
33	21	!	!	!	AND(I,X)	33
34	22	"	"	"		34
35	23	#	#	#		35
36	24	$	$	$	BIT Z	36
37	25	%	%	%	AND Z	37
38	26	&	&	&	ROL Z	38
39	27	'	'	'		39
40	28	(	(	(	PLP	40
41	29	)	)	)	AND #	41
42	2A	*	*	*	ROL A	42
43	2B	+	+	+		43
44	2C	,	,	,	BIT	44
45	2D	–	–	–	AND	45
46	2E	.	.	.	ROL	46
47	2F	/	/	/		47
48	30	0	0	0	BMI	48
49	31	1	1	1	AND(I),Y	49
50	32	2	2	2		50
51	33	3	3	3		51
52	34	4	4	4		52
53	35	5	5	5	AND Z,X	53
54	36	6	6	6	ROL Z,X	54
55	37	7	7	7		55
56	38	8	8	8	SEC	56
57	39	9	9	9	AND Y	57
58	3A	:	:	:		58
59	3B	;	;	;		59
60	3C	<	<	<		60
61	3D	=	=	=	AND X	61
62	3E	>	>	>	ROL X	62
63	3F	?	?	?		63
64	40	@	[graphic]	@	RTI	64
65	41	A	[graphic],a	A	EOR(I,X)	65
66	42	B	[graphic],b	B		66
67	43	C	[graphic],c	C		67
68	44	D	[graphic],d	D		68
69	45	E	[graphic],e	E	EOR Z	69
70	46	F	[graphic],f	F	LSR Z	70
71	47	G	[graphic],g	G		71
72	48	H	[graphic],h	H	PHA	72
73	49	I	[graphic],i	I	EOR #	73
74	4A	J	[graphic],j	J	LSR A	74
75	4B	K	[graphic],k	K		75
76	4C	L	[graphic],l	L	JMP	76
77	4D	M	[graphic],m	M	EOR	77
78	4E	N	[graphic],n	N	LSR	78
79	4F	O	[graphic],o	O		79
80	50	P	[graphic],p	P	BVC	80
81	51	Q	[graphic],q	Q	EOR(I),Y	81
82	52	R	[graphic],r	R		82
83	53	S	[graphic],s	S		83
84	54	T	[graphic],t	T		84
85	55	U	[graphic],u	U	EOR Z,X	85
86	56	V	[graphic],v	V	LSR Z,X	86
87	57	W	[graphic],w	W		87
88	58	X	[graphic],x	X	CLI	88
89	59	Y	[graphic],y	Y	EOR Y	89
90	5A	Z	[graphic],z	Z		90
91	5B	[	[graphic]	[		91
92	5C	\	[graphic]	\		92
93	5D	]	[graphic]	]	EOR X	93
94	5E	↑	π,[graphic]	↑	LSR X	94
95	5F	←	[graphic],[graphic]	←		95
96	60		[graphic]		RTS	96
97	61		[graphic]		ADC(I,X)	97
98	62		[graphic]			98
99	63		[graphic]			99
100	64		[graphic]			100
101	65		[graphic]		ADC Z	101
102	66		[graphic]		ROR Z	102
103	67		[graphic]			103
104	68		[graphic]		PLA	104
105	69		[graphic],[graphic]		ADC #	105
106	6A		[graphic]		ROR A	106
107	6B		[graphic]			107
108	6C		[graphic]		JMP(I)	108
109	6D		[graphic]		ADC	109
110	6E		[graphic]		ROR	110
111	6F		[graphic]			111
112	70		[graphic]		BVS	112
113	71		[graphic]		ADC(I),Y	113
114	72		[graphic]			114
115	73		[graphic]			115
116	74		[graphic]			116
117	75		[graphic]		ADC Z,X	117
118	76		[graphic]		ROR Z,X	118
119	77		[graphic]			119
120	78		[graphic]		SEI	120
121	79		[graphic]		ADC Y	121
122	7A		[graphic],[graphic]			122
123	7B		[graphic]			123
124	7C		[graphic]			124
125	7D		[graphic]		ADC X	125
126	7E		[graphic]		ROR X	126
127	7F		[graphic]			127

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
128	80		@	END		128
129	81		A	FOR	STA(I,X)	129
130	82		B	NEXT		130
131	83	load & run	C	DATA		131
132	84		D	INPUT#	STY Z	132
133	85		E	INPUT	STA Z	133
134	86		F	DIM	STX Z	134
135	87	bell	G	READ		135
136	88		H	LET	DEY	136
137	89	set/clr tab	I	GOTO		137
138	8A		J	RUN	TXA	138
139	8B		K	IF		139
140	8C		L	RESTORE	STY	140
141	8D	car ret	M	GOSUB	STA	141
142	8E	graphics	N	RETURN	STX	142
143	8F	bot right	O	REM		143
144	90		P	STOP	BCC	144
145	91	cur up	Q	ON	STA(I),Y	145
146	92	rvs off	R	WAIT		146
147	93	clear	S	LOAD		147
148	94	insert	T	SAVE	STY Z,X	148
149	95	ins line	U	VERIFY	STA Z,X	149
150	96	ers end	V	DEF	STX Z,Y	150
151	97		W	POKE		151
152	98		X	PRINT#	TYA	152
153	99	scroll up	Y	PRINT	STA Y	153
154	9A		Z	CONT	TXS	154
155	9B	escape	[	LIST		155
156	9C		\	CLR		156
157	9D	cur left	]	CMD	STA X	157
158	9E		↑	SYS		158
159	9F		←	OPEN		159
160	A0	□		CLOSE	LDY #	160
161	A1	▌	!	GET	LDA(I,X)	161
162	A2	▄	"	NEW	LDX #	162
163	A3	□	#	TAB(		163
164	A4	□	$	TO	LDY Z	164
165	A5	□	%	FN	LDA Z	165
166	A6	▒	&	SPC(	LDX Z	166
167	A7	□	'	THEN		167
168	A8	▄	(	NOT	TAY	168
169	A9	◩,▨	)	STEP	LDA #	169
170	AA	▐	*	+	TAX	170
171	AB	├	+	−		171
172	AC	▗	,	*	LDY	172
173	AD	└	-	/	LDA	173
174	AE	┐	.	↑	LDX	174
175	AF	▁	/	AND		175
176	B0	┌	0	OR	BCS	176
177	B1	┴	1	>	LDA(I),Y	177
178	B2	┬	2	=		178
179	B3	┤	3	<		179
180	B4	▎	4	SGN	LDY Z,X	180
181	B5	▍	5	INT	LDA Z,X	181
182	B6	▐	6	ABS	LDX Z,Y	182
183	B7	▔	7	USR		183
184	B8	▀	8	FRE	CLV	184
185	B9	▃	9	POS	LDA Y	185
186	BA	□,☑	:	SQR	TSX	186
187	BB	▖	;	RND		187
188	BC	▝	<	LOG	LDY X	188
189	BD	┘	=	EXP	LDA X	189
190	BE	▘	>	COS	LDX Y	190
191	BF	▚	?	SIN		191

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
192	C0	─		TAN	CPY #	192
193	C1	♠,a		ATN	CMP(I),X	193
194	C2	│,b		PEEK		194
195	C3	─,c		LEN		195
196	C4	─,d		STR$	CPY Z	196
197	C5	─,e		VAL	CMP Z	197
198	C6	─,f		ASC	DEC Z	198
199	C7	│,g		CHR$		199
200	C8	│,h		LEFT$	INY	200
201	C9	╮,i		RIGHT$	CMP #	201
202	CA	╰,j		MID$	DEX	202
203	CB	╯,k		GO		203
204	CC	└,l		CONCAT	CPY	204
205	CD	╲,m		DOPEN	CMP	205
206	CE	╱,n	Reverse of ASCII	DCLOSE	DEC	206
207	CF	┌,o		RECORD		207
208	D0	┐,p		HEADER	BNE	208
209	D1	●,q		COLLECT	CMP(I),Y	209
210	D2	─,r		BACKUP		210
211	D3	♥,s		COPY		211
212	D4	│,t		APPEND		212
213	D5	╭,u		DSAVE	CMP Z,X	213
214	D6	☒,v		DLOAD	DEC Z,X	214
215	D7	○,w		CATALOG		215
216	D8	♣,x		RENAME	CLD	216
217	D9	│,y		SCRATCH	CMP Y	217
218	DA	♦,z		DIRECTORY		218
219	DB	┼				219
220	DC	▌				220
221	DD	│			CMP X	221
222	DE	π,▒			DEC X	222
223	DF	◥,▧				223
224	E0		■		CPX #	224
225	E1		▐		SBC(I),X	225
226	E2		▀			226
227	E3		▆			227
228	E4		■		CPX Z	228
229	E5		▊		SBC Z	229
230	E6		▒		INC Z	230
231	E7		▉			231
232	E8		▒		INX	232
233	E9		◪,▨		SBC #	233
234	EA		▌		NOP	234
235	EB		┤			235
236	EC		▛		CPX	236
237	ED		┐		SBC	237
238	EE		└		INC	238
239	EF		▇			239
240	F0		┘		BEQ	240
241	F1		┬		SBC(I),Y	241
242	F2		┴			242
243	F3		├			243
244	F4		▊			244
245	F5		▋		SBC Z,X	245
246	F6		▌		INC Z,X	246
247	F7		▆			247
248	F8		▅		SED	248
249	F9		▀		SBC Y	249
250	FA		■,☑			250
251	FB		▜			251
252	FC		▛			252
253	FD		▟		SBC X	253
254	FE		▙		INC X	254
255	FF	π	▞	π		255

BASIC 2.0 / BASIC 4.0 Memory Map

Supplied by Jim Butterfield. Reference to DOS, MLM, 80-Column, or those marked with an * are for BASIC 4.0 only.

Hex	Dec	Description
0000 – 0002	0–2	USR jump
0003	3	Search character
0004	4	Scan-between-quotes flag
0005	5	Input buffer pointer; # of subscripts
0006	6	Default DIM flag
0007	7	Type: FF = string, 00 = numeric
0008	8	Type: 80 = integer, 00 = floating point
0009	9	Flag: DATA scan; LIST quote; memory
000A	10	Subscript flag; FNX flag
000B	11	0 = INPUT; $40 = GET; $98 = READ
000C	12	ATN sign/Comparison Evaluation flag
000D – 000F	13–15	Disk status DS$ descriptor
0010	16	Current I/O device for prompt-suppress
0011 – 0012	17–18	Integer value (for SYS, GOTO etc)
0013 – 0015	19–21	Pointers for descriptor stack
0016 – 001E	22–30	Descriptor stack(temp strings)
001F – 0022	31–34	Utility pointer area
0023 – 0027	35–39	Product area for multiplication
0028 – 0029	40–41	**Pointer:** Start of BASIC
002A – 002B	42–43	**Pointer:** Start of Variables
002C – 002D	44–45	**Pointer:** Start of Arrays
002E – 002F	46–47	**Pointer:** End of Arrays
0030 – 0031	48–49	**Pointer:** String Storage (moving down)
0032 – 0033	50–51	**Pointer:** Utility String
0034 – 0035	52–53	**Pointer:** Limit of Memory
0036 – 0037	54–55	Current BASIC line number
0038 – 0039	56–57	Previous BASIC line number
003A – 003B	58–59	**Pointer:** BASIC statement for CONT
003C – 003D	60–61	Current DATA line number
003E – 003F	62–63	Current DATA address
0040 – 0041	64–65	Input vector
0042 – 0043	66–67	Current variable name
0044 – 0045	68–69	Current variable address
0046 – 0047	70–71	Variable pointer for FOR/NEXT
0048 – 0049	72–73	Y-save; op-save; BASIC pointer save
004A	74	Comparison symbol accumulator
004B – 0050	75–80	Misc work area, pointers, etc
0051 – 0053	81–83	Jump vector for functions
0054 – 005D	84–93	Misc numeric work area
005E	94	Accum#1: Exponent
005F – 0062	95–98	Accum#1: Mantissa
0063	99	Accum#1: Sign
0064	100	Series evaluation constant pointer
0065	101	Accum#1 hi-order (overflow)
0066 – 006B	102–107	Accum#2: Exponent, etc
006C	108	Sign comparison, Acc#1 vs #2
006D	106	Accum#1 lo-order (rounding)
006E – 006F	110–111	Cassette buff len/Series pointer
0070 – 0087	112–135	CHRGET subroutine; get BASIC char
0077 – 0078	119–120	BASIC pointer (within subrtn)
0088 – 008C	136–140	Random number seed.
008D – 008F	141–143	Jiffy clock for TI and TI$
0090 – 0091	144–145	Hardware interrupt vector
0092 – 0093	146–147	BRK interrupt vector
0094 – 0095	148–149	NMI interrupt vector
0096	150	Status word ST
0097	151	Which key down; 255 = no key
0098	152	Shift key: 1 if depressed
0099 – 009A	153–154	Correction clock
009B	155	Keyswitch PIA: STOP and RVS flags
009C	156	Timing constant for tape
009D	157	Load = 0, Verify = 1
009E	158	Number of characters in keybd buffer
009F	159	Screen reverse flag
00A0	160	IEEE output; 255 = character pending
00A1	161	End-of-line-for-input pointer
00A3 – 00A4	163–164	Cursor log (row, column)
00A5	165	IEEE output buffer
00A6	166	Key image
00A7	167	0 = flash cursor
00A8	168	Cursor timing countdown
00A9	169	Character under cursor
00AA	170	Cursor in blink phase
00AB	171	EOT received from tape
00AC	172	Input from screen/from keyboard
00AD	173	X save
00AE	174	How many open files
00AF	175	Input device, normally 0
00B0	176	Output CMD device, normally 3
00B1	177	Tape character parity
00B2	178	Byte received flag
00B3	179	Logical Address temporary save
00B4	180	Tape buffer character/MLM command
00B5	181	File name pointer/MLM flag, counter
00B7	183	Serial bit count
00B9	185	Cycle counter
00BA	186	Tape writer countdown
00BB – 00BC	187–188	Tape buffer pointers, #1 and #2
00BD	189	Write leader count; read pass1/2
00BE	190	Write new byte; read error flag
00BF	191	Write start bit; read bit seq error
00C0 – 00C1	192–193	Error log pointers, pass1/2
00C2	194	0 = Scan/1–15 = Count/$40 = Load/$80 = End
00C3	195	Write leader length; read checksum
00C4 – 00C5	196–197	Pointer to screen line
00C6	198	Position of cursor on above line
00C7 – 00C8	199–200	Utility pointer: tape, scroll
00C9 – 00CA	201–202	Tape end addrs/End of current program
00CB – 00CC	203–204	Tape timing constants
00CD	205	0 = direct cursor, else programmed
00CE	206	Tape read timer 1 enabled
00CF	207	EOT received from tape
00D0	208	Read character error
00D1	209	# characters in file name
00D2	210	Current file logical address
00D3	211	Current file secondary addrs
00D4	212	Current file device number
00D5	213	Right-hand window or line margin
00D6 – 00D7	214–215	**Pointer:** Start of Tape Buffer
00D8	216	Line where cursor lives
00D9	217	Last key/checksum/misc.
00DA – 00DB	218–219	File name pointer
00DC	220	Number of INSERTs outstanding
00DD	221	Write shift word/read character in
00DE	222	Tape blocks remaining to write/read
00DF	223	Serial word buffer
00E0 – 00F8	224–248	(40-column) Screen line wrap table
00E0 – 00E1	224–225	(80-column) Top, bottom of window
00E2	226	(80-column) Left window margin
00E3	227	(80-column) Limit of keybd buffer
00E4	228	(80-column) Key repeat flag
00E5	229	(80-column) Repeat countdown
00E6	230	(80-column) New key marker
00E7	231	(80-column) Chime time
00E8	232	(80-column) HOME count
00E9 – 00EA	233–234	(80-column) Input vector
00EB – 00EC	235–236	(80-column) Output vector
00F9 – 00FA	249–250	Cassette status, #1 and #2
00FB – 00FC	251–252	Tape start address/MLM Pointer
00FD – 00FE	253–254	MLM/DOS pointer/misc.
0100 – 010A	256–266	STR$ work area/MLM work
0100 – 013E	256–318	Tape read error log
0100 – 01FF	256–511	Processor stack
0200 – 0250	512–592	MLM work area; Input buffer
0251 – 025A	593–602	File logical address table
025B – 0264	603–612	File device number table
0265 – 026E	613–622	File secondary addr table
026F – 0278	623–632	Keyboard input buffer
027A – 0339	634–825	Tape#1 input buffer
033A – 03F9	826–1017	Tape#2 input buffer
033A	826	DOS character pointer
033B	827	DOS drive 1 flag
033C	828	DOS drive 2 flag
033D	829	DOS length/write flag
033E	830	DOS syntax flags
033F – 0340	831–832	DOS disk ID
0341	833	DOS command string count
0342 – 0352	834–850	DOS file name buffer
0353 – 0380	851–896	DOS command string buffer
03EE – 03F7	1006–1015	(80-column) Tab stop table
03FA – 03FB	1018–1019	Monitor extension vector
03FC	1020	IEEE timeout defeat* $FF-disable
0400 – 7FFF	1024–32767	Available RAM including expansion
8000 – 83FF	3276833791	(40-column) Video RAM
8000 – 87FF	3276834815	(80-column) Video RAM
9000 – AFFF	3686445055	Available ROM expansion area* (2.0: –BFFF, –49151)
B000 – DFFF	4505657343	BASIC, DOS, Machine Lang Monitor (2.0 BASIC: C000–E0F8, 49152–57592)
E000 – E7FF	5734459391	Screen, Keyboard, Interrupt programs (2.0: E0F9–)
E810 – E813	5940859411	PIA 1 – Keyboard I/O
E820 – E823	5942459427	PIA 2 – IEEE-488 I/O
E840 – E84F	5945659471	VIA – I/O and timers
E880 – E881	5952059521	(80-column) CRT Controller
F000 – FFFF	6144065535	Reset, I/O handlers, Tape routines

E810	Diagnostic Sense	IEEE EOI In	Cassette Sense #2	#1	Keyboard Row Select PA				59408
E811	Tape #1 Input Flag		EOI Out CA2			DDRA Access	Cassette #1 Read Control CA1		59409
E812	Keyboard Row Input								59410
E813	Retrace I Flag		Cassette #1 Motor Output CB2			DDRB Access	Retrace Interrupt Control CB1		59411
E820	IEEE Input								59424
E821	ATN I Flag		IEEE NDAC Out CA2			DDRA Access	IEEE ATN In Control CA1		59425
E822	IEEE Output								59426
E823	SRQ I Flag		IEEE DAV Out CB2			DDRB Access	IEEE SRQ In Control CB1		59427
E840	DAV In	NRFD In	Retrace In	Cass. #2 Motor	Cassette Output	ATN Out	NRFD Out	NDAC In PB	59456
E841	Parallel User Port (PUP.) I/O with Handshake								59457
E842	Data Direction Register B (for E840)								59458
E843	Data Direction Register A (for E84F, PUP.)								59459
E844	Timer 1 L								59460
E845	Timer 1 H								59461
E846	Timer 1 Latch L								59462
E847	Timer 1 Latch H								59463
E848	Timer 2 L								59464
E849	Timer 2 H								59465
E84A	Shift Register								59466
E84B	T1 Control PB7 Out		T2 Ctrl PB6 Sense	Shift Register Control			PB, PA Latch Control		59467
E84C	CB2 (PUP. Pin M) Control In/Out			CB1 In Cassette #2 Polarity	CA2 (Graphics, Lower Case) In/Out			CA1 In Polarity	59468
E84D	IRQ Status	T1 INT	T2 INT	CB1 Cass #2 INT	CB2 INT	SRQ INT	CA1 (PUP.B) INT	CA2 INT	59469
E84E	Enable Clear/Set	T1 INT Enab	T2 INT Enab	CB1 INT Enab	CB2 INT Enab	SRQ INT Enab	CA1 INT Enab	CA2 INT Enab	59470
E84F	Parallel User Port I/O (PA) PA								59471

BASIC 2.0 / BASIC 4.0 ROM Routines

The BASIC 4.0 40–character and 80–character machines are the same except for addresses $E000–$E7FF. This map shows where various routines lie. The first address is not necessarily the proper entry point for the routine. Similarly, many routines require register setup or data preparation before calling.

BASIC 2.0 ROM Routines

Address	Description
C000 – C045	Action addresses for primary keywords
C046 – C073	Action addresses for functions
C074 – C091	Hierarchy & action addrs for operators
C092 – C192	Table of BASIC keywords
C193 – C2A9	BASIC messages, mostly error msgs
C2AA – C2D7	Search stack FOR/GOSUB
C2D8 – C31A	Open up space in memory
C31B – C327	Test: stack too deep?
C328 – C354	Check available memory
C355	Send canned error message, then.
C389 – C3AA	Warm start (ready.)
C3AB – C441	Handle new BASIC line input
C442 – C46E	Rebuild chaining of BASIC lines
C46F – C494	Receive line from keyboard
C495 – C52B	Crunch keywords into BASIC tokens
C52C – C55A	Search BASIC for given line number
C55B	Perform [NEW], and,
C577 – C5A6	Perform [CLR]
C5A7 – C5B4	Reset BASIC execution to start
C5B5 – C657	Perform [LIST]
C658 – C6FF	Perform [FOR]
C700 – C72F	Execute BASIC statement
C730 – C73E	Perform [RESTORE]
C73F – C76A	Perform [STOP] or [END]
C76B – C784	Perform [CONT]
C785 – C78F	Perform [RUN]
C790 – C7AC	Perform [GOSUB]
C7AD – C7D9	Perform [GOTO]
C7DA	Perform [RETURN], then.
C7F3 – C80D	Perform [DATA]: skip statement
C80E	Scan for next BASIC statement
C811 – C82F	Scan for next BASIC line
C830	Perform [IF], and perhaps:
C843 – C852	Perform [REM]: skip line
C853 – C872	Perform [ON]
C873 – C8AC	Accept fixed–point number
C8AD – C98A	Perform [LET]
C98B – C990	Perform [PRINT#]
C991 – C9A4	Perform [CMD]
C9A5 – CA1B	Perform [PRINT]
CA1C – CA38	Print string from memory
CA39 – CA4E	Print single format character
CA4F – CA7C	Handle bad input data
CA7D – CAA6	Perform [GET]
CAA7 – CAC0	Perform [INPUT#]
CAC1 – CAF9	Perform [INPUT]
CAFA – CB06	Prompt and receive input
CB07 – CBFB	Perform [READ]
CBFC – CC1F	Canned Input error messages
CC20 – CC78	Perform [NEXT]
CC79 – CC9E	Check type mismatch
CC9F	Evaluate expression
CDEC	Evaluate expr. within ( )
CDF2 – CE02	Check parenthesis, comma
CE03 – CE07	Syntax error exit
CE08 – CE88	Variable name setup
CE89 – CEC7	Set up function references
CEC8 – CEF7	Perform [OR], [AND]
CEF8 – CF5F	Perform comparisons
CF60 – CF6C	Perform [DIM]
CF6D – CFF6	Search for variable
CFF7 – D077	Create new variable
D078 – D088	Setup array pointer
D089 – D08C	32768 in floating binary
D08D – D0AB	Evaluate integer expression
D0AC – D258	Find or make array
D259	Perform [FRE], and:
D26D – D279	Convert fixed–to–floating
D27A – D27F	Perform [POS]
D280 – D28C	Check not Direct
D28D – D2BA	Perform [DEF]
D2BB – D2CD	Check FNx syntax
D2CE – D33E	Evaluate FNx
D33F – D34E	Perform [STR$]
D34F – D360	Do string vector
D361 – D3CD	Scan, set up string
D3CE – D3FF	Allocate space for string
D400 – D516	Garbage collection
D517 – D553	Concatenate
D554 – D57C	Store string
D57D – D5B4	Discard unwanted string
D5B5 – D5C5	Clean descriptor stack
D5C6 – D5D9	Perform [CHR$]
D5DA – D605	Perform [LEFT$]
D606 – D610	Perform [RIGHT$]
D611 – D63A	Perform [MID$]
D63B – D655	Pull string data
D656 – D65B	Perform [LEN]
D65C – D664	Switch string to numeric
D665 – D674	Perform [ASC]
D675 – D686	Get byte parameter
D687 – D6C5	Perform [VAL]
D6C6 – D6D1	Parameters for POKE/WAIT
D6D2 – D6E7	Convert floating–to–fixed
D6E8 – D706	Perform [PEEK]
D707 – D70F	Perform [POKE]
D710 – D72B	Perform [WAIT]
D72C – D732	Add 0.5
D733 – D744	Perform subtraction
D745 – D76D	Microsoft Joke (WAIT 6502)
D76E – D852	Perform addition
D853 – D889	Complement accum#1
D88A – D88E	Overflow exit
D88F – D8C7	Multiply–a–byte
D8C8 – D8f5	Constants
D8F6	Perform [LOG]
D937 – D997	Perform multiplication
D998 – D9C2	Unpack memory into accum#2
D9C3 – D9DF	Test & adjust accumulators
D9E0 – D9ED	Handle overflow and underflow
D9EE – DA04	Multiply by 10
DA05 – DA09	10 in floating binary
DA0A	Divide by 10
DA13	Perform divide–by
DA1E – DAAD	Perform divide–into
DAAE – DAD2	Unpack memory into accum#1
DAD3 – DB07	Pack accum#1 into memory
DB08 – DB17	Move accum#2 to #1
DB18 – DB26	Move accum#1 to #2
DB27 – DB36	Round accum#1
DB37 – DB44	Get accum#1 sign
DB45 – DB63	Perform [SGN]
DB64 – DB66	Perform [ABS]
DB67 – DBA6	Compare accum#1 to memory
DBA7 – DBD7	Floating–to–fixed
DBD8 – DBFE	Perform [INT]
DBFF – DC89	Convert string to floating–point
DC8A – DCBE	Get new ASCII digit
DCBF – DCCD	Constants
DCCE	Print IN, then:
DCD9 – DCE8	Print BASIC line #
DCE9 – DE1C	Convert floating–point to ASCII
DE1D – DE5D	Constants
DE5E	Perform [SQR]
DE68	Perform power function
DEA1 – DEAB	Perform negation
DEAC – DED9	Constants
DEDA – DF2C	Perform [EXP]
DF2D – DF76	Series evaluation
DF77 – DF7E	RND constants
DF7F – DFD7	Perform [RND]
DFD8	Perform [COS]
DFDF – E027	Perform [SIN]
E028 – E053	Perform [TAN]
E054 – E08B	Constants
E08C – E0BB	Perform [ATN]
E0BC – E0F8	Constants
E0F9 – E110	CHRGET sub for zero page
E111 – E115	Initial RND seed
E116 – E1B6	BASIC cold start
E1B7 – E1DD	Power up msg, " bytes free"
E1DE	Init I/O regs and:
E229	Clear screen and:
E257 – E284	Home cursor
E285 – E33E	Input from screen or keyboard
E33F – E34B	Test for quote; test quote flag
E34C – E38A	Set screen print parameters
E38B – E395	Prevent 80–char line getting longer
E396 – E3B3	Turn 40 char line into 80 char line
E3B4 – E3D7	Back into previous line
E3D8 – E518	Handle ASCII char for screen output
E519 – E53E	Go to next screen line
E53F – E5B9	Scroll screen
E5BA – E61A	Open a line on screen
E61B – E62D	Main interrupt entry
E62E – E6E9	Interrupt: clock, cursor, keyboard
E6EA – E6F7	Output character
E6F8 – E769	Table: keyboard matrix decoder
E76A – E796	MLM sub: output hex digits
E797 – E7A6	MLM sub: swap TMP0 and TMP2
E7A7 – E7F6	MLM sub: input hex digits
E7F7 – E7FF	MLM sub: print '?'
F000 – F0B5	File messages
F0B6 – F127	Send 'Talk', 'Listen', IEEE command
F128 – F135	Send char to IEEE
F136 – F155	Write Timeout, Device Not Present
F156 – F163	Send canned I/O message
F164 – F16E	Send 'Listen', secondary address
F16F – F17E	Send normal (deferred) IEEE char
F17F – F18B	Drop IEEE device
F18C – F1D0	Input byte from IEEE
F1D1 – F1E0	GET a byte
F1E1 – F231	INPUT a byte
F232 – F26D	Output a byte
F26E	Abort files
F284 – F28C	Restore default I/O devices
F28D – F2A8	Find/setup file data
F2A9 – F300	Perform [CLOSE]
F301 – F30E	Test STOP key
F30F – F314	Action STOP key
F315 – F31C	Send message if Direct mode
F31D – F321	Test if Direct mode
F322 – F3C1	Program load subroutine
F3C2 – F409	Perform [LOAD]
F40A – F43D	Print Searching, Loading, Verifying
F43E – F45F	Get Load/Save parameters
F460 – F465	Get a byte parameter
F466 – F493	Send filename to IEEE
F494 – F4B6	Find specific tape header
F4B7 – F4CD	Perform [VERIFY]
F4CE – F50D	Get Open/Close parameters
F50E – F515	Abort if end–of–line
F516 – F520	Check comma, else Syntax Error
F521 – F5A5	Perform [OPEN]
F5A6 – F5D9	Find any tape header
F5DA – F63B	Write tape header
F63C – F655	Get start/end addrs from header
F656 – F66B	Set buffer address
F66C – F683	Set buffer start & end addrs
F684 – F68C	Perform [SYS]
F68D – F69D	Set tape write start & end
F69E – F728	Perform [SAVE]
F729 – F76C	Update clock
F76D – F76F	TI constant, modulus 24 hrs
F770 – F7BB	Connect input device
F7BC – F805	Connect output device
F806 – F811	Bump tape buffer pointer
F812 – F834	Wait for PLAY
F835 – F846	Test cassette switches
F847 – F854	Wait for PLAY AND RECORD
F855	Initiate tape read
F886 – F8E5	Initiate tape write
F8E6 – F8EF	Test I/0 complete
F8F0 – F8FF	Test STOP key
F900 – F930	Tape bit timing adjust
F931	Read tape bits
FA57 – FB75	Store tape characters
FB76 – FB7E	Reset tape read address
FB7F – FB83	Flag error into ST
FB84 – FB92	Reset counters for new byte
FB93 – FBAE	Write a bit to tape
FBAF – FC40	Tape write
FC41 – FC7A	Write tape leader
FC7B – FC95	Terminate tape, restore interrupt
FC96 – FCA5	Set interrupt vector
FCA6 – FCB3	Turn off tape motor
FCB4 – FCC5	Checksum calculation
FCC6 – FCD0	Advance load/save pointer
FCD1 – FCFD	Power–on Reset
FCFE – FD10	Table of interrupt vectors
FD11 – FFB0	Machine Language Monitor
FFB1 – FFBF	CBM copyright statement
Jump Table:	
FFC0	OPEN
FFC3	CLOSE
FFC6	Set input device
FFC9	Set output device
FFCC	Restore default I/O devices
FFCF	INPUT a byte
FFD2	Output a byte
FFD5	LOAD
FFD8	SAVE
FFDB	VERIFY
FFDE	SYS
FFE1	Test stop key
FFE4	GET byte
FFE7	Abort all files
FFEA	Update clock
FFED – FFF9	Unused
FFFA – FFFF	Hard vectors: NMI, Reset, IRQ

BASIC 4.0 ROM Routines

Address	Description
B000 – B065	Action addresses for primary keywords
B066 – B093	Action addresses for functions
B094 – B0B1	Hierarchy & action addrs for operators
B0B2 – B20C	Table of BASIC keywords
B20D – B321	BASIC messages, mostly error msgs
B322 – B34F	Search stack for FOR/GOSUB
B350 – B392	Open up space in memory
B393 – B39F	Test: stack too deep?
B3A0 – B3CC	Check available memory
B3CD	Send canned error message, then:
B3FF – B41E	Warm start; wait for BASIC command
B41F – B4B5	Handle new BASIC line input
B4B6 – B4E1	Rebuild chaining of BASIC lines
B4E2 – B4FA	Receive line from keyboard
B4FB – B5A2	Crunch keywords into BASIC tokens
B5A3 – B5D1	Search BASIC for given line number
B5D2	Perform [NEW], and:
B5EC – B621	Perform [CLR]
B622 – B62F	Reset BASIC execution to start
B630 – B6DD	Perform [LIST]
B6DE – B784	Perform [FOR]
B785 – B7B6	Execute BASIC statement
B7B7 – B7C5	Perform [RESTORE]
B7C6 – B7ED	Perform [STOP] or [END]
B7EE – B807	Perform [CONT]
B808 – B812	Perform [RUN]
B813 – B82F	Perform [GOSUB]
B830 – B85C	Perform [GOTO]
B85D	Perform [RETURN], then.
B883 – B890	Perform [DATA]: skip statement
B891	Scan for next BASIC statement
B894 – B8B2	Scan for next BASIC line
B8B3	Perform [IF], and perhaps:
B8C6 – B8D5	Perform [REM]: skip line
B8D6 – B8F5	Perform [ON]
B8F6 – B92F	Accept fixed–point number
B930 – BA87	Perform [LET]
BA88 – BA8D	Perform [PRINT#]
BA8E – BAA1	Perform [CMD]
BAA2 – BB1C	Perform [PRINT]
BB1D – BB39	Print string from memory
BB3A – BB4B	Print single format character
BB4C – BB79	Handle bad input data
BB7A – BBA3	Perform [GET]
BBA4 – BBBD	Perform [INPUT#]
BBBE – BBF4	Perform [INPUT]
BBF5 – BC01	Prompt and receive input
BC02 – BCF6	Perform [READ]
BCF7 – BD18	Canned Input error messages
BD19 – BD71	Perform [NEXT]
BD72 – BD97	Check type mismatch
BD98	Evaluate expression
BEE9	Evaluate expr. within parentheses
BEEF	Check parenthesis, comma
BF00 – BF0B	Syntax error exit
BF8C – C046	Variable name setup
C047 – C085	Set up function references
C086 – C0B5	Perform [OR], [AND]
C0B6 – C11D	Perform comparisons
C11E – C12A	Perform [DIM]
C12B – C1BF	Search for variable
C1C0 – C2C7	Create new variable
C2C8 – C2D8	Setup array pointer
C2D9 – C2DC	32768 in floating binary
C2DD – C2FB	Evaluate integer expression
C2FC – C4A7	Find or make array
C4A8	Perform [FRE], and:
C4BC – C4C8	Convert fixed–to–floating
C4C9 – C4CE	Perform [POS]
C4CF – C4DB	Check not Direct
C4DC – C509	Perform [DEF]
C50A – C51C	Check FNx syntax
C51D – C58D	Evaluate FNx
C58E – C59D	Perform [STR$]
C59E – C5AF	Do string vector
C5B0 – C61C	Scan, set up string
C61D – C669	Allocate space for string
C66A – C74E	Garbage collection
C74F – C78B	Concatenate
C78C – C7B4	Store string
C7B5 – C810	Discard unwanted string
C811 – C821	Clean descriptor stack
C822 – C835	Perform [CHR$]
C836 – C861	Perform [LEFT$]
C862 – C86C	Perform [RIGHT$]
C86D – C896	Perform [MID$]
C897 – C8B1	Pull string data
C8B2 – C8B7	Perform [LEN]
C8B8 – C8C0	Switch string to numeric
C8C1 – C8D0	Perform [ASC]
C8D1 – C8E2	Get byte parameter
C8E3 – C920	Perform [VAL]
C921 – C92C	Parameters for POKE/WAIT
C92D – C942	Convert floating–to–fixed
C943 – C959	Perform [PEEK]
C95A – C962	Perform [POKE]
C963 – C97E	Perform [WAIT]
C97F – C985	Add 0.5
C986	Perform subtraction
C998 – CA7C	Perform addition
CA7D – CAB3	Complement accum#1
CAB4 – CAB8	Overflow exit
CAB9 – CAF1	Multiply–a–byte
CAF2 – CB1F	Constants
CB20	Perform [LOG]
CB5E – CBC1	Perform multiplication
CBC2 – CBEC	Unpack mem. into accum#2
CBED – CC09	Test & adjust accumulators
CC0A – CC17	Handle overflow & underflow
CC18 – CC2E	Multiply by 10
CC2F – CC33	10 in floating binary
CC34	Divide by 10
CC3D	Perform divide–by
CC45 – CCD7	Perform divide–into
CCD8 – CCFC	Unpack mem. into accum#1
CCFD – CD31	Pack accum#1 into memory
CD32 – CD41	Move accum#2 to #1
CD42 – CD50	Move accum#1 to #2
CD51 – CD60	Round accum#1
CD61 – CD6E	Get accum#1 sign
CD6F – CD8D	Perform [SGN]
CD8E – CD90	Perform [ABS]
CD91 – CDD0	Compare accum#1 to memory
CDD1 – CE01	Floating–to–fixed
CE02 – CE28	Perform [INT]
CE29 – CEB3	Convert string to floating–pt
CEB4 – CEE8	Get new ASCII digit
CEE9 – CEF8	Constants
CF78	Print IN, then
CF7F – CF92	Print BASIC line #
CF93 – D0C6	Convert floating–pt to ASCII
D0C7 – D107	Constants
D108	Perform [SQR]
D112	Perform power function
D14B – D155	Perform negation
D156 – D183	Constants
D184 – D1D6	Perform [EXP]
D1D7 – D220	Series evaluation
D221 – D228	RND constants
D229 – D281	Perform [RND]
D282	Perform [COS]
D289 – D2D1	Perform [SIN]
D2D2 – D2FD	Perform [TAN]
D2FE – D32B	Constants
D32C – D35B	Perform [ATN]
D35C – D398	Constants
D399 – D3B5	CHRGET sub for zero page
D3B6 – D471	BASIC cold start
D472 – D716	Machine Language Monitor
D717 – D7AB	MLM subroutines
D7AC – D802	Perform [RECORD]
D803 – D837	Disk parameter checks
D838 – D872	Dummy disk control msgs
D873 – D919	[CATALOG] or [DIRECTORY]
D91A – D92E	Output
D92F – D941	Find spare secondary address
D942 – D976	Perform [DOPEN]
D977 – D990	Perform [APPEND]
D991 – D9D1	Get disk status
D9D2 – DA06	Perform [HEADER]
DA07 – DA30	Perform [DCLOSE]
DA31 – DA64	Set up disk record
DA65 – DA7D	Perform [COLLECT]
DA7E – DAA6	Perform [BACKUP]
DAA7 – DAC6	Perform [COPY]
DAC7 – DAD3	Perform [CONCAT]
DAD4 – DB0C	Insert command string values
DB0D – DB39	Perform [DSAVE]
DB3A – DB65	Perform [DLOAD]
DB66 – DB98	Perform [SCRATCH]
DB99 – DB9D	Check Direct command
DB9E – DBD6	Query ARE YOU SURE?
DBD7 – DBE0	Print BAD DISK
DBE1 – DBF9	Clear DS$ and ST
DBFA – DC67	Assemble disk command string
DC68 – DE29	Parse BASIC DOS command
DE2C – DE48	Get Device number
DE49 – DE86	Get file name
DE87 – DE9C	Get small variable parameter
Entry points only for E000–E7FF	
E000	Register/screen initialization
E0A7	Input from keyboard
E116	Input from screen
E202	Output character
E442	Main Interrupt entry
E455	Interrupt: clock, cursor, keyboard
E600	Exit from Interrupt
F000 – F0D1	File messages
F0D2	Send 'Talk'
F0D5	Send 'Listen'
F0D7	Send IEEE command character
F109 – F142	Send byte to IEEE
F143 – F150	Send byte and clear ATN
F151 – F16B	Option: timeout or wait
F16C – F16F	DEVICE NOT PRESENT
F170 – F184	Timeout on read, clear control lines
F185 – F192	Send canned file message
F193 – F19D	Send byte, clear control lines
F19E – F1AD	Send normal (deferred) IEEE char
F1AE – F1BF	Drop IEEE device
F1C0 – F204	Input byte from IEEE
F205 – F214	GET a byte
F215 – F265	INPUT a byte
F266 – F2A1	Output a byte
F2A2	Abort files
F2A6 – F2C0	Restore default I/O devices
F2C1 – F2DC	Find/setup file data
F2DD – F334	Perform [CLOSE]
F335 – F342	Test STOP key
F343 – F348	Action STOP key
F349 – F350	Send message if Direct mode
F351 – F355	Test if Direct mode
F356 – F400	Program load subroutine
F401 – F448	Perform [LOAD]
F449 – F46C	Print SEARCHING
F46D – F47C	Print LOADING or VERIFYING
F47D – F4A4	Get Load/Save parameters
F4A5 – F4D2	Send name to IEEE
F4D3 – F4F5	Find specific tape header
F4F6 – F50C	Perform [VERIFY]
F50D – F55F	Get Open/Close parameters
F560 – F5E4	Perform [OPEN]
F5E5 – F618	Find any tape header
F619 – F67A	Write tape header
F67B – F694	Get start/end addrs from header
F695 – F6AA	Set buffer address
F6AB – F6C2	Set buffer start & end addrs
F6C3 – F6CB	Perform [SYS]
F6CC – F6DC	Set tape write start & end
F6DD – F767	Perform [SAVE]
F768 – F7AE	Update clock
F7AF – F7FD	Connect input device
F7FE – F84A	Connect output device
F84B – F856	Bump tape buffer pointer
F857 – F879	Wait for PLAY
F87A – F88B	Test cassette switch
F88C – F899	Wait for RECORD
F89A	Initiate tape read
F8CB	Initiate tape write
F8E0 – F92A	Common tape I/O
F92B – F934	Test I/0 complete
F935 – F944	Test STOP key
F945 – F975	Tape bit timing adjust
F976 – FA9B	Read tape bits
FA9C – FBBA	Read tape characters
FBBB – FBC3	Reset tape read address
FBC4 – FBC8	Flag error into ST
FBC9 – FBD7	Reset counters for new byte
FBD8 – FBF3	Write a bit to tape
FBF4 – FC85	Tape write
FC86 – FCBF	Write tape leader
FCC0 – FCDA	Terminate tape; restore interrupt
FCDB – FCEA	Set interrupt vector
FCEB – FCF8	Turn off tape motor
FCF9 – FD0A	Checksum calculation
FD0B – FD15	Advance load/save pointer
FD16 – FD4B	Power–on Reset
FD4C – FD5C	Table of interrupt vectors
Jump Table:	
FF93 – FF9E	CONCAT,DOPEN,DCLOSE,RECORD
FF9F – FFAA	HEADER,COLLECT,BACKUP,COPY
FFAB – FFB6	APPEND,DSAVE,DLOAD,CATALOG
FFB7 – FFBC	RENAME,SCRATCH
FFBD	Get disk status
FFC0	OPEN
FFC3	CLOSE
FFC6	Set input device
FFC9	Set output device
FFCC	Restore default I/O devices
FFCF	INPUT a byte
FFD2	Output a byte
FFD5	LOAD
FFD8	SAVE
FFDB	VERIFY
FFDE	SYS
FFE1	Test stop key
FFE4	GET byte
FFE7	Abort all files
FFEA	Update clock
FFED – FFF9	unused
FFFA – FFFF	Hard vectors: NMI, Reset, IRQ

BASIC 2.0 / BASIC 4.0 Memory Map

With Zero Page Contents at Power-Up

Reference to DOS, MLM, 80-Column, or those marked with an * are for BASIC 4.0 only.
There are some differences between the 40 and 80-column machines. BASIC 2.0 Zero Page contents are mostly identical except for vectors.

Location				Contents				Description
Hex		Dec		4000		8000		
				Hex	Dec	Hex	Dec	
00 -02	00	0–2	0	4C	76	4C	76	USR jump instruction
	01		1	73	115	73	115	;JMP $C373
	02		2	C3	195	C3	195	
03	03	3	3	22	34	22	34	Search character
04	04	4	4	00	0	00	0	Scan–between–quotes flag
05	05	5	5	5B	91	5B	91	Input buffer pointer
06	06	6	6	00	0	FF	255	Default DIM flag
07	07	7	7	00	0	00	0	Type: $FF = string, $00 = numeric
08	08	8	8	00	0	00	0	Type: $80 = integer, 00 = floating pt
09	09	9	9	04	4	04	4	Flag: DATA scan; LIST quote; memory
0A	0A	10	10	00	0	00	0	Subscript flag; FNx flag
0B	0B	11	11	00	0	00	0	0 = INPUT; $40 = GET; $98 = READ
0C	0C	12	12	00	0	FF	255	ATN sign/comparison evaluation flag
0D -0F	0D	13–15	13	00	0	00	0	Disk status DS$ descriptor
	0F		14	FF	255	FF	255	
	0F		15	00	0	00	0	
10	10	16	16	00	0	00	0	Current I/O prompt flag
11 -12	11	17–18	17	72	114	72	114	Integer value (for SYS, GOTO etc.)
	12		18	D4	212	D4	212	
13 -15	13	19–21	19	16	22	16	22	Pointers for descriptor stack
	14		20	13	19	13	19	
	15		21	00	0	00	0	
16 -1E	16	22–30	22	08	8	08	8	Descriptor stack (temporary strings)
	17		23	12	18	12	18	
	18		24	B3	179	B3	179	
	19		25	00	0	00	0	
	1A		26	FF	255	FF	255	
	1B		27	00	0	00	0	
	1C		28	FF	255	FF	255	
	1D		29	00	0	00	0	
	1E		30	FF	255	FF	255	
1F -22	1F	31–34	31	40	64	40	64	Utility pointer area
	20		32	B2	178	B2	178	
	21		33	E9	233	E9	233	
	22		34	CE	206	CE	206	
23 -27	23	35–39	35	48	72	00	0	Product area for multiplication
	24		36	00	0	FF	255	
	25		37	00	0	00	0	
	26		38	00	0	FF	255	
	27		39	00	0	00	0	
28 -29	28	40–41	40	01	1	01	1	**Pointer:** Start of BASIC
	29		41	04	4	04	4	;0401
2A -2B	2A	42–43	42	03	3	03	3	**Pointer:** Start of Variables
	2B		43	04	4	04	4	;0403
2C -2D	2C	44–45	44	03	3	03	3	**Pointer:** Start of Arrays
	2D		45	04	4	04	4	;0403
2E -2F	2E	46–47	46	03	3	03	3	**Pointer:** End of Arrays
	2F		47	04	4	04	4	;0403
30 -31	30	48–49	48	00	0	00	00	**Pointer:** String Storage (moving down)
	31		49	80	128	80	128	;8000
32 -33	32	50–51	50	FE	254	FF	255	**Pointer:** String Utility
	33		51	7F	127	00	0	
34 -35	34	52–53	52	00	0	00	0	**Pointer:** Limit of Memory
	35		53	80	128	80	128	;8000
36 -37	36	54–55	54	14	20	FF	255	Current BASIC line number
	37		55	FF	255	FF	255	
38 -39	38	56–57	56	00	0	FF	255	Previous BASIC line number
	39		57	80	128	00	0	
3A -3B	3A	58–59	58	01	1	FF	255	**Pointer:** BASIC statement for CONT
	3B		59	00	0	00	0	
3C -3D	3C	60–61	60	00	0	FF	255	Current DATA line number
	3D		61	50	80	00	0	
3E -3F	3E	62–63	62	00	0	00	0	Current data address
	3F		63	04	4	04	4	
40 -41	40	64–65	64	00	0	FF	255	Input vector
	41		65	00	0	00	0	
42 -43	42	66–67	66	04	4	FF	255	Current variable name
	43		67	00	0	00	0	
44 -45	44	68–69	68	24	36	24	36	Current variable address
	45		69	04	4	00	0	
46 -47	46	70–71	70	82	130	FF	255	Variable pointer for FOR/NEXT
	47		71	04	4	00	0	
48 -49	48	72–73	72	FF	255	FF	255	Y–save; op–save; BASIC pointer save
	49		73	00	0	00	0	
4A	4A	74	74	00	0	00	0	Comparison symbol accumulator
4B -50	4B	75–80	75	00	0	00	0	Miscellaneous work area, pointers, etc.
	4C		76	FF	255	FF	255	
	4D		77	16	22	00	0	
	4E		78	00	0	FF	255	
	4F		79	00	0	00	0	
	50		80	03	3	03	3	
51 -53	51	81–83	81	4C	76	4C	76	Jump vector for functions
	52		82	43	67	FF	255	
	53		83	00	0	00	0	
54 -5D	54	84–93	84	FF	255	FF	255	Miscellaneous numeric work area
	55		85	87	135	00	0	
	56		86	04	4	FF	255	
	57		87	80	128	00	0	
	58		88	03	3	FF	255	
	59		89	00	0	00	0	
	5A		90	00	0	00	0	
	5B		91	00	0	00	0	
	5C		92	00	0	00	0	
	5D		93	00	0	00	0	
5E	5E	94	94	90	144	94	144	Accum*1: Exponent
5F -62	5F	95–98	95	00	0	00	0	Accum*1: Mantissa
	60		96	00	0	00	0	
	61		97	D4	212	D4	212	
	62		98	72	114	72	114	
63	63	99	99	00	0	00	0	Accum*1: Sign
64	64	100	100	00	0	00	0	Series evaluation constant pointer
65	65	101	101	00	0	00	0	Accum*1 hi–order (overflow)
66 -6B	66	102–107	102	90	144	90	144	Accum*2: Exponent
	67		103	D4	212	D4	212	Accum*2: Mantissa
	68		104	6C	108	6C	108	
	69		105	00	0	00	0	
	6A		106	00	0	00	0	
	6B		107	00	0	00	0	Accum*2: Sign
6C	6C	108	108	00	0	00	0	Sign comparison, Acc*1 vs *2
6D	6D	109	109	00	0	00	0	Accum*1 lo–order (rounding)
6E -6F	6E	110–111	110	0A	10	0A	10	Cassette buff len/series pointer
	6F		111	B3	179	B3	179	
70 -87	70	112–135	112	E6	230	E6	230	CHRGET subroutine; get BASIC char
	71		113	77	119	77	119	;INC $77
	72		114	D0	208	D0	208	;BNE $0076
	73		115	02	2	02	2	
	74		116	E6	230	E6	230	;INC $78
	75		117	78	120	78	120	
	76		118	AD	173	AD	173	;LDA $0202
	77		119	02	2	02	2	
	78		120	02	2	02	2	
	79		121	C9	201	C9	201	;CMP *$3A
	7A		122	3A	58	3A	58	
	7B		123	B0	176	B0	176	;BCS $0087
	7C		124	0A	10	0A	10	
	7D		125	C9	201	C9	201	;CMP *$20
	7E		126	20	32	20	32	
	7F		127	F0	240	F0	240	;BEQ $0070
	80		128	EF	239	EF	239	
	81		129	38	56	38	56	;SEC
	82		130	E9	233	E9	233	;SBC *$30
	83		131	30	48	30	48	
	84		132	38	56	38	56	;SEC
	85		133	E9	233	E9	233	;SBC *$D0
	86		134	D0	208	D0	208	
	87		135	60	96	60	96	;RTS
77 -78	77	119–120	119	02	2	02	2	BASIC pointer (within subroutine)
	78		120	02	2	02	2	
88 -8C	88	136–140	136	80	128	80	128	Random number seed
	89		137	4F	79	4F	79	
	8A		138	C7	199	C7	199	
	8B		139	52	82	52	82	
	8C		140	F4	244	FF	255	
8D -8F	8D	141–143	141	00	0	00	0	Jiffy clock for TI and TI$
	8E		142	15	21	08	8	
	8F		143	89	137	1F	31	
90 -91	90	144–145	144	55	85	55	85	Hardware interrupt vector IRQ
	91		145	E4	228	E4	228	
92 -93	92	146–147	146	78	120	78	120	BRK interrupt vector
	93		147	D4	212	D4	212	
94 -95	94	148–149	148	FF	255	FF	255	NMI interrupt vector
	95		149	B3	179	B3	179	

Location				Contents				Description
Hex		Dec		4000 Hex	4000 Dec	8000 Hex	8000 Dec	
96	96	150	150	00	0	00	0	Status word ST
97	97	151	151	FF	255	FF	25	Which key down: 255 = no key
98	98	152	152	00	0	00	0	Shift key: 1 if depressed
99-9A	99	153-154	153	19	25	D5	213	Correction clock
	9A		154	02	2	00	0	
9B	9B	155	155	FF	255	FF	255	Keyswitch PIA: STOP and RVS flags
9C	9C	156	156	00	0	00	0	Timing constant for tape
9D	9D	157	157	00	0	00	0	Load = 0, Verify = 1
9E	9E	158	158	00	0	00	0	# of chars in keyboard buffer
9F	9F	159	159	00	0	00	0	Screen reverse flag
A0	A0	160	160	FF	255	FF	255	IEEE output; 255 = character pending
A1	A1	161	161	1E	30	20	32	End-of-line-for-input pointer
A2	A2	162	162	00	0	00	0	Not used
A3-A4	A3	163-164	163	0A	10	0A	10	Cursor log (row, column)
	A4		164	1E	30	20	32	
A5	A5	165	165	1E	30	20	32	IEEE output buffer
A6	A6	166	166	FF	255	FF	255	Key image
A7	A7	167	167	01	1	01	1	0 = flash cursor
A8	A8	168	168	02	2	02	2	Cursor timing countdown
A9	A9	169	169	20	32	20	32	Character under cursor
AA	AA	170	170	00	0	00	0	Cursor in blink phase
AB	AB	171	171	00	0	00	0	EOT received from tape
AC	AC	172	172	00	0	00	0	Input from screen/from keyboard
AD	AD	173	173	00	0	00	0	X save
AE	AE	174	174	00	0	00	0	How many open files
AF	AF	175	175	00	0	00	0	Input device, normally 0
B0	B0	176	176	03	3	03	3	Output CMD device, normally 3
B1	B1	177	177	00	0	00	0	Tape character parity
B2	B2	178	178	00	0	00	0	Byte received flag
B3	B3	179	179	00	0	00	0	Logical address temporary save
B4	B4	180	180	07	7	07	7	Tape buffer character; MLM command
B5	B5	181	181	00	0	00	0	File name pointer; MLM flag, counter
B6	B6	182	182	00	0	00	0	Function not known
B7	B7	183	183	00	0	00	0	Serial bit count
B8	B8	184	184	00	0	00	0	Unused
B9	B9	185	185	00	0	00	0	Cycle counter
BA	BA	186	186	00	0	00	0	Tape writer countdown
BB-BC	BB	187-188	187	00	0	00	0	Tape buffer pointers, #1 and #2
	BC		188	00	0	00	0	
BD	BD	189	189	00	0	00	0	Write leader count; read pass1/2
BE	BE	190	190	00	0	00	0	Write new byte; read error flag
BF	BF	191	191	00	0	00	0	Write start bit; read bit seq error
C0-C1	C0	192-193	192	00	0	00	0	Error log pointers, pass1/2
	C1		193	00	0	00	0	
C2	C2	194	194	00	0	00	0	0 = scan/1-15 = count/$40 = load $80 = end
C3	C3	195	195	00	0	00	0	Write leader length; read checksum
C4-C5	C4	196-197	196	90	144	20	32	Pointer to screen line
	C5		197	81	129	83	131	
C6	C6	198	198	1E	31	21	33	Position of cursor on above line
C7-C8	C7	199-200	199	C7	199	C7	199	Utility pointer; tape, scroll
	C8		200	00	0	00	0	
C9-CA	C9	201-202	201	00	0	24	36	Tape end addrs/end of current prog
	CA		202	01	1	10	16	
CB-CC	CB	203-204	203	00	0	00	0	Tape timing constants
	CC		204	00	0	00	0	
CD	CD	205	205	00	0	00	0	0 = direct cursor, else programmed
CE	CE	206	206	00	0	00	0	Tape read timer 1 = enabled
CF	CF	207	207	00	0	00	0	EOT received from tape
D0	D0	208	208	00	0	00	0	Read character error
D1	D1	209	209	0D	13	0F	15	# characters in file name
D2	D2	210	210	00	0	00	0	Current file logical address
D3	D3	211	211	61	97	61	97	Current file secondary address
D4	D4	212	212	08	8	08	8	Current file device number
D5	D5	213	213	27	39	4F	79	Right-hand window or line margin
D6-D7	D6	214-215	214	00	0	00	0	**Pointer:** Start of Tape Buffer

Location				Contents				Description
Hex		Dec		4000 Hex	4000 Dec	8000 Hex	8000 Dec	
	D7		215	00	0	00	0	
D8	D8	216	216	0A	10	0A	10	Line where cursor lives
D9	D9	217	217	0D	13	0D	13	Last key/checksum/misc.
DA-DB	DA	218-219	218	09	9	09	9	File name pointers
	DB		219	02	2	02	2	
DC	DC	220	220	00	0	00	0	Number of INSERTs outstanding
DD	DD	221	221	00	0	00	0	Write shift word/read character in
DE	DE	222	222	00	0	00	0	Tape blocks remaining to write/read
DF	DF	223	223	00	0	00	0	Serial word buffer
E0-F8	E0	224-248	224	80	128			(40-column) screen wrap table
	E1		225	80	128			
	E2		226	80	128			
	E3		227	80	128			
	E4		228	80	128			
	E5		229	80	128			
	E6		230	80	128			
	E7		231	81	129			
	E8		232	81	129			
	E9		233	81	129			
	EA		234	81	129			
	EB		235	81	129			
	EC		236	81	129			
	ED		237	82	130			
	EE		238	82	130			
	EF		239	82	130			
	F0		240	82	130			
	F1		241	82	130			
	F2		242	82	130			
	F3		243	82	130			
	F4		244	83	131			
	F5		245	83	131			
	F6		246	83	131			
	F7		247	83	131			
	F8		248	83	131			
E0	E0	224	224			00	0	(80 column) Screen top window
E1	E1	225	225			18	24	(80 column) Screen bottom window
E2	E2	226	226			00	0	(80 column) Left window margin
E3	E3	227	227			09	9	(80 column) Limit of keyboard buffer
E4	E4	228	228			00	0	(80 column) Key repeat flag
E5	E5	229	229			0E	14	(80 column) Repeat countdown
E6	E6	230	230			10	16	(80 column) New key marker
E7	E7	231	231			10	16	(80 column) Chime time
E8	E8	232	232			00	0	(80 column) HOME count
E9-EA	E9	233-234	233			1D	29	(80 column) Input vector
	EA		234			E1	225	
EB-EC	EB	235-236	235			0C	12	(80 column) Output vector
	EC		236			E2	226	
ED-F7	ED	237-247	237			00	0	(80 column) Not used
	EE		238			00	0	
	EF		239			00	0	
	F0		240			00	0	
	F1		241			00	0	
	F2		242			00	0	
	F3		243			00	0	
	F4		244			00	0	
	F5		245			00	0	
	F6		246			00	0	
	F7		247			00	0	
F8	F8	248	248			00	0	(80 column) Counter to speed TI by 6/5
F9-FA	F9	249-250	249	00	0	00	0	Cassette status, #1 and #2
	FA		250	00	0	00	0	
FB-FC	FB	251-252	251	00	0	00	0	MLM pointer/tape start address
	FC		252	00	0	00	0	
FD-FE	FD	253-254	253	00	0	24	36	MLM, DOS pointer, misc.
	FE		254	01	1	10	16	
FF	FF	255	255	00	0	00	0	Unused

0100-010A	256-266	STR$ work area/MLM work
0100-013E	256-318	Tape read error log
0100-01FF	256-511	Processor stack
0200-0250	512-592	MLM work area; Input buffer
0251-025A	593-602	File logical address table
025B-0264	603-612	File device number table
0265-026E	613-622	File secondary addr table
026F-0278	623-632	Keyboard input buffer
027A-0339	634-825	Tape#1 input buffer
033A-03F9	826-1017	Tape#2 input buffer
033A	826	DOS character pointer
033B	827	DOS drive 1 flag
033C	828	DOS drive 2 flag
033D	829	DOS length/write flag
033E	830	DOS syntax flags
033F-0340	831-832	DOS disk ID
0341	833	DOS command string count
0342-0352	834-850	DOS file name buffer

0353-0380	851-896	DOS command string buffer
03EE-03F7	1006-1015	(80-column) Tab stop table
03FA-03FB	1018-1019	Monitor extension vector
03FC	1020	IEEE timeout defeat* $FF - disable
0400-7FFF	1024-32767	Available RAM including expansion
8000-83FF	32768-33791	(40-column) Video RAM
8000-87FF	32768-34815	(80-column) Video RAM
9000-AFFF	36864-45055	Available ROM expansion area* (2.0: -BFFF, -49151)
B000-DFFF	45056-57343	Basic, DOS, Machine Lang Monitor (2.0: Basic, C000-E0F8, 49152-57592)
E000-E7FF	57344-59391	Screen, Keyboard, Interrupt programs (2.0: E0F9-)
E810-E813	59408-59411	PIA 1 - Keyboard I/O
E820-E823	59424-59427	PIA 2 - IEEE-488 I/O
E840-E84F	59456-59471	VIA - I/O and timers
E880-E881	59520-59521	(80-column) CRT Controller
F000-FFFF	61440-65535	Reset, I/O handlers, Tape routines

VIC 20 Memory Map

Address	Decimal	Description
0000 -0002	0-2	USR jump
0003 -0004	3-4	Float-Fixed vector
0005 -0006	5-6	Fixed-Float vector
0007	7	Search character
0008	8	Scan-quotes flag
0009	9	TAB column save
000A	10	0 = LOAD, 1 = VERIFY
000B	11	Input buffer pointer/# subscrpt
000C	12	Default DIM flag
000D	13	Type: FF = string, 00 = numeric
000E	14	Type: 80 = integer, 00 = floating point
000F	15	DATA scan/LIST quote/memry flag
0010	16	Subscript/FNx flag
0011	17	0 = INPUT;$40 = GET;$98 = READ
0012	18	ATN sign/Comparison eval flag
0013	19	Current I/O prompt flag
0014 -0015	20-21	Integer value
0016	22	**Pointer:** temporary strg stack
0017 -0018	23-24	Last temp string vector
0019 -0021	25-33	Stack for temporary strings
0022 -0025	34-37	Utility pointer area
0026 -002A	38-42	Product area for multiplication
002B -002C	43-44	**Pointer:** Start of BASIC
002D-002E	45-46	**Pointer:** Start of Variables
002F -0030	47-48	**Pointer:** Start of Arrays
0031 -0032	49-50	**Pointer:** End of Arrays
0033 -0034	51-52	**Pointer:** String Storage (moving down)
0035 -0036	53-54	**Pointer:** Utility String
0037 -0038	55-56	**Pointer:** Limit of Memory
0039 -003A	57-58	Current BASIC line number
003B -003C	59-60	Previous BASIC line number
003D-003E	61-62	**Pointer:** BASIC statement for CONT
003F -0040	63-64	Current DATA line number
0041 -0042	65-66	Current DATA address
0043 -0044	67-68	Input vector
0045 -0046	69-70	Current variable name
0047 -0048	71-72	Current variable address
0049 -004A	73-74	Variable pointer for FOR/NEXT
004B -004C	75-76	Y-save op-save; BASIC pointer save
004D	77	Comparison symbol accumulator
004E -0053	78-83	Misc work area, pointers, etc
0054 -0056	84-86	Jump vector for functions
0057 -0060	87-96	Misc numeric work area
0061	97	Accum#1: Exponent
0062 -0065	98-101	Accum#1: Mantissa
0066	102	Accum#1: Sign
0067	103	Series evaluation constant pointer
0068	104	Accum#1 hi-order (overflow)
0069	105	Accum#2: Exponent
006A -006D	106-109	Accum#2 Mantissa
006E	110	Accum#2: Sign
006F	111	Sign comparison, Acc#1 vs #2
0070	112	Accum#1 lo-order (rounding)
0071 -0072	113-114	Cassette buff len/Series pointer
0073 -008A	115-138	CHRGET subroutine; get BASIC char
007A -007B	122-123	BASIC pointer (within subrtn)
008B -008F	139-143	RND seed value
0090	144	Status word ST
0091	145	Keyswitch PIA: STOP and RVS flags
0092	146	Timing constant for tape
0093	147	Load = 0, Verify = 1
0094	148	Serial output: deferred char flag
0095	149	Serial deferred character
0096	150	Tape EOT received
0097	151	Register save
0098	152	How many open files
0099	153	Input device, normally 0
009A	154	Output CMD device, normally 3
009B	155	Tape character parity
009C	156	Byte-received flag
009D	157	Direct = $80/RUN = 0 output control
009E	158	Tp Pass 1 error log/char buffer
009F	159	Tp Pass 2 err log corrected
00A0 -00A2	160-162	Jiffy Clock HML
00A3	163	Serial bit count/EOI flag
00A4	164	Cycle count
00A5	165	Countdown,tape write/bit count
00A6	166	Tape buffer pointer
00A7	167	Tp Wrt ldr count/Rd pass/inbit
00A8	168	Tp Wrt new byte/Rd error/inbit cnt
00A9	169	Wrt start bit/Rd bit err/stbit
00AA	170	Tp Scan;Cnt;Ld;End/byte assy
00AB	171	Wr lead length/Rd checksum/parity
00AC-00AD	172-173	**Pointer:** tape bufr, scrolling
00AE-00AF	174-175	Tape end adds/End of program
00B0 -00B1	176-177	Tape timing constants
00B2 -00B3	178-179	**Pointer:** Start of Tape Buffer
00B4	180	1 = Tp timer enabled, bit cnt
00B5	181	Tp EOT/RS232 next bit to send
00B6	182	Read character error/outbyte buf
00B7	183	# characters in file name
00B8	184	Current logical file
00B9	185	Current secndy address
00BA	186	Current device
00BB-00BC	187-188	Pointer to file name
00BD	189	Wr shift word/Rd input char
00BE	190	# blocks remaining to Wr/Rd
00BF	191	Serial word buffer
00C0	192	Tape motor interlock
00C1 -00C2	193-194	I/O start adds
00C3 -00C4	195-196	Kernel setup pointer
00C5	197	Last key pressed
00C6	198	# chars in keybd buffer
00C7	199	Screen reverse flag
00C8	200	End-of-line for input pointer
00C9 -00CA	201-202	Input cursor log (row, column)
00CB	203	Which key: 64 if no key
00CC	204	0 = flash cursor
00CD	205	Cursor timing countdown
00CE	206	Character under cursor
00CF	207	Cursor in blink phase
00D0	208	Input from screen/from keyboard
00D1 -00D2	209-210	Pointer to screen line
00D3	211	Position of cursor on above line
00D4	212	0 = direct cursor, else programmed
00D5	213	Current screen line length
00D6	214	Row where curosr lives
00D7	215	Last inkey/checksum/buffer
00D8	216	# of INSERTs outstanding
00D9 -00F0	217-240	Screen line link table
00F1	241	Dummy screen link
00F2	242	Screen row marker
00F3 -00F4	243-244	Screen color pointer
00F5 -00F6	245-246	Keyboard pointer
00F7 -00F8	247-248	RS-232 Rcv pntr
00F9 -00FA	249-250	RS-232 Tx pntr
00FF -010A	256-266	Floating to ASCII work area
0100 -013E	256-318	Tape error log
0100 -01FF	256-511	Processor stack area
0200 -0258	512-600	BASIC input buffer
0259 -0262	601-610	Logical file table
0263 -026C	611-620	Device # table
026D-0276	621-630	Sec Adds table
0277 -0280	631-640	Keybd buffer
0281 -0282	641-642	Start of BASIC Memory
0283 -0284	643-644	Top of BASIC Memory
0285	645	Serial bus timeout flag
0286	646	Current colour code
0287	647	Colour under cursor
0288	648	Screen memory page
0289	649	Max size of keybd buffer
028A	650	Repeat all keys
028B	651	Repeat speed counter
028C	652	Repeat delay counter
028D	653	Keyboard Shift/Control flag
028E	654	Last shift pattern
028F -0290	655-656	Keyboard table settup pointer
0291	657	Keymode (Kattacanna)
0292	658	0 = scroll enable
0293	659	VIC chip control
0294	660	VIC chip command
0295 -0296	661-662	Bit timing
0297	663	RS-232 status
0298	664	# bits to send
0299 -029A	665-666	RS-232 speed/code
029B	667	RS232 receive pointer
029C	668	RS232 input pointer
029D	669	RS232 transmit pointer
029E	670	RS232 output pointer
029F -02A0	671-672	IRQ save during tape I/O
0300 -0301	768-769	Error message link
0302 -0303	770-771	BASIC warm start link
0304 -0305	772-773	Crunch BASIC tokens link
0306 -0307	774-775	Print tokens link
0308 -0309	776-777	Start new BASIC code link
030A -030B	778-779	Get arithmetic element link
030C -0313	780-787	Unused
0314 -0315	788-789	Hardware interrupt vector (EABF)
0316 -0317	790-791	Break interrupt vector (FED2)
0318 -0319	792-793	NMI interrupt vector (FEAD)
031A -031B	794-795	OPEN vector (F40A)
031C -031D	796-797	CLOSE vector (F34A)
031E -031F	798-799	Set-input vector (F2C7)
0320 -0321	800-801	Set-output vector (F309)
0322 -0323	802-803	Restore I/O vector (F3F3)
0324 -0325	804-805	INPUT vector (F20E)
0326 -0327	806-807	Output vector (F27A)
0328 -0329	808-809	Test-STOP vector (F770)
032A -032B	810-811	GET vector (F1F5)
032C -032D	812-813	Abort I/O vector (F3EF)
032E -032F	814-815	USR vector (FED2)
0330 -0331	816-817	LOAD link (F549)
0332 -0333	818-819	SAVE link (F685)
033C -03FB	828-1019	Cassette buffer
03FC -03FF	1020-1023	Unused
0400 -0FFF	1024-4095	3K RAM expansion area
1000 -1DFF	4096-7679	Normal BASIC memory
1E00 -1FF9	7680-8191	Normal Screen memory
1000 -11F9	4096-4601	Screen memory w/expansion
1200 -	4608-	BASIC memory w/expansion
2000 -7FFF	8192-32767	Memory expansion area
8000 -8FFF	32768-36863	Character bit maps
9000 -900F	36864-36879	Video Interface Chip
9110 -912F	37136-37151	VIA Interface – NMI
9120 -912F	37152-37167	VIA Interface – IRQ
9400 -95FF	37888-38399	Alternate Colour Nybble area
9600 -97FF	38400-38911	Main Colour Nybble area
A000 -BFFF	40960-49151	Plug-in ROM area
C000 -FFFF	49152-65535	ROM BASIC and Operating System
FF8A-FFF5	65418-65525	Jump Table, Including.
FFC6		Set Input channel
FFC9		Set Output channel
FFCC		Restore default I/O channels
FFCF		INPUT
FFD2		PRINT
FFE1		Test Stop key
FFE4		GET

VIC 20 ROM Routines

Address	Routine
C000	ROM control vectors
C00C	Keyword action vectors
C052	Function vectors
C080	Operator vectors
C09E	Keywords
C19E	Error messages
C328	Error message vectors
C365	Miscellaneous messages
C38A	Scan stack for FOR/GOSUB
C3B8	Move memory
C3FB	Check stack depth
C408	Check memory space
C435	'OUT OF MEMORY'
C437	Error routine
C469	Break entry
C474	'READY.'
C480	Ready for BASIC
C49C	Handle new line
C533	Re-chain lines
C560	Receive input line
C579	Crunch tokens
C613	Find BASIC line
C642	Perform [NEW]
C65E	Perform [CLR]
C68E	Back up text pointer
C69C	Perform [LIST]
C742	Perform [FOR]
C7ED	Execute statement
C81D	Perform [RESTORE]
C82C	Break
C82F	Perform [STOP]
C831	Perform [END]
C857	Perform [CONT]
C871	Perform [RUN]
C883	Perform [GOSUB]
C8A0	Perform [GOTO]
C8D2	Perform [RETURN]
C8F8	Perform [DATA]
C906	Scan for next statement
C928	Perform [IF]
C93B	Perform [REM]
C94B	Perform [ON]
C96B	Get fixed point number
C9A5	Perform [LET]
CA80	Perform [PRINT#]
CA86	Perform [CMD]
CAA0	Perform [PRINT]
CB1E	Print message from (y,a)
CB3B	Print format character
CB4D	Bad-input routines
CB7B	Perform [GET]
CBA5	Perform [INPUT#]
CBBF	Perform [INPUT]
CBF9	Prompt & input
CC06	Perform [READ]
CCFC	Input error messages
CD1E	Perform [NEXT]
CD78	Type-match check
CD9E	Evaluate expression
CEA8	Constant – PI
CEF1	Evaluate within brackets
CEF7	Check for ')'
CEFF	Check for comma
CF08	Syntax error
CF14	Check range
CF28	Search for variable
CFA7	Set up FN reference
CFE6	Perform [OR]
CFE9	Perform [AND]
D016	Compare
D081	Perform [DIM]
D08B	Locate variable
D113	Check alphabetic
D11D	Create variable
D194	Array pointer subroutine
D1A5	Value 32768
D1B2	Float-fixed conversion
D1D1	Set up array
D245	'BAD SUBSCRIPT'
D248	'ILLEGAL QUANTITY'
D34C	Compute array size
D37D	Perform [FRE]
D391	Fixed-float conversion
D39E	Perform [POS]
D3A6	Check direct
D3B3	Perform [DEF]
D3E1	Check FN syntax
D3F4	Perform [FN]
D465	Perform [STR$]
D475	Calculate string vector
D487	Set up string
D4F4	Make room for string
D526	Garbage collection
D5BD	Check salvageability
D606	Collect string
D63D	Concatenate
D67A	Build string to memory
D6A3	Discard unwanted string
D6DB	Clean descriptor stack
D6EC	Perform [CHR$]
D700	Perform [LEFT$]
D72C	Perform [RIGHT$]
D737	Perform [MID$]
D761	Pull string parameters
D77C	Perform [LEN]
D782	Exit string-mode
D78B	Perform [ASC]
D79B	Input byte parameter
D7AD	Perform [VAL]
D7EB	Get params for POKE/WAIT
D7F7	Float-fixed
D80D	Perform [PEEK]
D824	Perform [POKE]
D82D	Perform [WAIT]
D849	Add 0.5
D850	Subtract-from
D853	Perform [SUBTRACT]
D86A	Perform [ADD]
D947	Complement fac#1
D97E	'OVERFLOW'
D983	Multiply by zero byte
D9EA	Perform [LOG]
DA2B	Perform [MULTIPLY]
DA59	Multiply-a-bit
DA8C	Memory to FAC#2
DAB7	Adjust FAC#1/#2
DAD4	Underflow/overflow
DAE2	Multiply by 10
DAF9	+ 10 in floating pt
DAFE	Divide by 10
DB12	Perform [DIVIDE]
DBA2	Memory to fac#1
DBC7	FAC#1 to memory
DBFC	FAC#2 to fac#1
DC0C	FAC#1 to FAC#2
DC1B	Round FAC#1
DC2B	Get sign
DC39	Perform [SGN]
DC58	Perform [ABS]
DC5B	Compare FAC#1 to mem
DC9B	Float-fixed
DCCC	Perform [INT]
DCF3	String to fac
DD7E	Get ASCII digit
DDDD	Float to ASCII
DF16	Decimal constants
DF3A	TI constants
DF71	Perform [SQR]
DF7B	Perform [POWER]
DFB4	Perform [NEGATIVE]
DFED	Perform [EXP]
E040	Series evaluate 1
E056	Series evaluate 2
E094	Perform [RND]
E0f6	?? Breakpoints ??
E127	Perform [SYS]
E153	Perform [SAVE]
E162	Perform [VERIFY]
E165	Perform [LOAD]
E1BB	Perform [OPEN]
E1C4	Perform [CLOSE]
E1D1	Parameters for LOAD/SAVE
E203	Check default parameters
E20B	Check for comma
E216	Parameters for open/close
E261	Perform [COS]
E268	Perform [SIN]
E2b1	Perform [TAN]
E30B	Perform [ATN]
E378	Initialize
E387	CHRGET for zero page
E3A4	Initialize BASIC
E429	Power-up message
E44F	Vectors for $300
E45B	Initialize vectors
E467	Warm restart
E476	Program patch area
E4A0	Serial output '1'
E4A9	Serial output '0'
E4B2	Get serial input & clock
E4BC	Program patch area
E500	Set 6522 addrs
E505	Set screen limits
E50A	Track cursor location
E518	Initalize I/O
E54C	Normalize screen
E55F	Clear screen
E581	Home cursor
E587	Set screen pointers
E5BB	Set I/O defaults
E5C3	Set VIC chip defaults
E5CF	Input from keyboard
E64F	Input from screen
E6B8	Quote mark test
E6C5	Set up screen print
E6EA	Advance cursor
E715	Retreat cursor
E72D	Back into previous line
E742	Output to screen
E8C3	Go to next line
E8D8	Do 'RETURN'
E8E8	Check line decrement
E8FA	Check line increment
E912	Set colour code
E921	Colour code table
E929	Code conversion
E975	Scroll screen
E9EE	Open space on screen
EA56	Move screen line
EA6E	Synch colour transfer
EA7E	Set start-of-line
EA8D	Clear screen line
EAA1	Print to screen
EAAA	Store on screen
EAB2	Synch colour to char
EABF	Interrupt (IRQ)
EB1E	Check keyboard
EC00	Set text mode
EC46	Keyboard vectors
EC5E	Keyboard maps
ED21	Graphics/text control
ED30	Set graphics mode
ED5B	Wrap up screen line
ED6A	Shifted key matrix
EDA3	Control key matrix
EDE4	VIC chip defaults
EDFD	Screen line adds low
EE14	Send 'talk'
EE17	Send 'listen'
EE1C	Send control char
EE49	Send to serial bus
EEB7	Timeout on serial
EEC0	Send listen SA
EEC5	Clear ATN
EECE	Send talk SA
EEE4	Send serial deferred
EEF6	Send 'untalk'
EF04	Send 'unlisten'
EF19	Receive from serial bus
EF84	Clock line on
EF8D	Clock line off
EF96	Delay 1 ms
EFA3	RS232 send (NMI)
EFEE	New RS232 byte send
F016	Error or quit
F027	Compute bit count
F036	RS232 receive (NMI)
F05B	Setup to receive
F09D	Receive parity error
F0A2	Receive overrun error
F0A5	Receive break error
F0A8	Receive frame error
F0B9	Bad device
F0BC	File to RS232
F0ED	Send to RS232 buffer
F116	Input from RS232 buffer
F14F	Get from RS232 buffer
F160	Check serial bus idle
F174	Messages
F1E2	Print if direct
F1F5	Get ..
F205	.. from RS232
F20E	Input
F250	Get .. tape/serial/RS232
F27A	Output..
F290	.. to tape
F2C7	Set input device
F309	Set output device
F34A	Close
F3CF	Find file
F3DF	Set file values
F3EF	Abort all files
F3F3	Restore default I/O
F40A	Do file opening
F495	Send SA
F4C7	Open RS232
F542	LOAD program
F647	'SEARCHING'
F659	Print file name
F66A	'LOADING/VERIFYING'
F675	SAVE program
F728	'SAVING'
F734	Bump clock
F760	Get time
F767	Set time
F770	Action stop key
F77E	File Error Messages
F7AF	Find any tape header
F7E7	Write tape header
F84D	Get buffer address
F854	Set buffer start, end pointers
F867	Find specific header
F88A	Bump tape pointer
F894	'PRESS PLAY ..'
F8AB	Check cassette status
F8B7	'PRESS RECORD ..'
F8C0	Initiate tape read
F8E3	Initiate tape write
F8F4	Common tape read/write
F94B	Check tape stop
F95D	Set timing
F98E	Read bits (IRQ)
FAAD	Store characters
FBD2	Reset pointer
FBDB	New tape character setup
FBEA	Toggle tape
FC06	Data write
FC0B	Tape write (IRQ)
FC95	Leader write (IRQ)
FCCF	Restore vectors
FCF6	Set vector
FD08	Kill motor
FD11	Check read/write pointer
FD1B	Bump read/write pointer
FD22	Powerup entry
FD3F	Check A-ROM
FD52	Set Kernal2
FD8D	Initialize system constants
FDF1	IRQ vectors
FDF9	Initialize I/O regs
FE49	Save data name
FE50	Save file details
FE57	Get status
FE66	Flag ST
FE6F	Set timeout
FE73	Read/set top of memory
FE82	Read/set bottom of memory
FE91	Test memory location
FEA9	NMI interrupt entry
FED2	RESET/STOP warm start
FEDE	NMI RS232 sequences
FF56	Restore & exit
FF5C	RS232 timing table
FF72	Main IRQ entry
FF8A	Jumbo jump table
FFFA	Hardware vectors

VIC 20 Standard Configuration

Hex	Contents	Decimal
FFFF		65535
	8K Kernal ROM	
E000		57344
	8K BASIC ROM	
C000		49152
A000		40960
95FF		38399
	Colour Nybble Area	
9600		38400
	VIC Chip & I/O	
9000		36864
	Character Set	
8000		32768
2000		4096
	½K Screen RAM from basic VIC 20	
1E00		7680
	3½ K RAM for BASIC	
1000		4096
0400		1024
	1K RAM Work Space	
0000		0

VIC 20 Expansion RAM Memory Changes

Exp RAM at:	BASIC Text	Screen	Colour Table
none	4096 / $1000	7680 / $1E00	38400 / $9600
1024 / 4095*	1024 / $0400	7680 / $1E00	38400 / $9600
8192 and up	4608 / $1200	4096 / $1000	37888 / $9400

* VIC 1210 3K RAM Expander

VIC 20 With 40K RAM

VIC 1020 Expansion Module Required with:
1 - VIC 1210 3K RAM
2 - VIC 1110 8K RAM (Switches 2,3,4 down - Switch 1 up)
3 - VIC 1110 8K RAM (Switches 1,3,4 down - Switch 2 up)
4 - VIC 1111 16K RAM

Hex	Contents	Decimal
FFFF		65535
	8K Kernal ROM	
E000		57344
	8K BASIC ROM	
C000		49152
	VIC 1110 8K RAM (2) (usable only with PEEK, POKE & M/L)	
A000		40960
95FF		38399
	Colour Nybble Area	
9400		37888
	VIC Chip & I/O	
9000		36864
	Character Set	
8000		32768
	VIC 1110 8K RAM (3)	27½ K for BASIC
	VIC 1111 16K RAM (4)	
	3½ K of RAM from basic VIC 20	
1200		4608
	½K Screen RAM from basic VIC 20	
1000		4096
	VIC 1210 3K RAM (1) (usable only with PEEK, POKE & M/L)	
0400		1024
	1K RAM Work Space	
0000		0

6560 VIC Chip

Hex	Contents			Decimal
9000	Interlace	Left Margin (= 5)		36864
9001	Top Margin (= 25)			36865
9002	Screen Ad Bit 9	Number of Columns (= 22)		36866
9003	Bit 0	Number of Rows (= 23)	Double Char	36867
9004	Input Raster Value: Bits 1-8			36868
9005	Screen Address Bits 13-10	Character Address Bits 13-10		36869
9006	Light Pen Input	Horizontal		36870
9007		Vertical		36871
9008	Paddle Input	X		36872
9009		Y		36873
900A	ON	Voice 1 Frequency		36874
900B	ON	Voice 2 Frequency		36875
900C	ON	Voice 3 Frequency		36876
900D	ON	Noise Frequency		36877
900E	Multi Colour Mode	Sound Amplitude		36878
900F	Background Colour	Foregnd/ Backgnd	Border Colour	36879

6522 VIA 1

Hex									Decimal
9110	DSR In	CTS In		DCD* In	RI* In	DTR Out	RTS Out	Data In	37136
	RS-232 Interface or Parallel User Port								
9111	*Unused - see $911F								37137
9112	Data Direction Register B (for $9110)								37138
9113	Data Direction Register A (for $911F)								37139
9114	T1-L	RS 232 Send Speed;							37140
9115	T1-H	Tape Write Timing							37141
9116	T1-Latch L								37142
9117	T1 Latch H								37143
9118	T2-L	RS 232 Input Timing							37144
9119	T2-H								37145
911A	Shift Register (* unused)								37146
911B	T1 Control		T2 Ctrl	Shift Register Control			PB LE	PA LE	37147
911C	CB2: RS 232 Send			CB1 Ctrl	CA2: Tape Motor Ctrl			CA1 Ctrl	37148
911D	NMI:	T1	T2	CB1: RS 232 In			CA1: RESTORE		37149
911E	NMI En.	T1 Enab	T2 Enab	CB1 En.			CA1 En.		37150
911F	ATN Out	Tape Sense	Fire	Joystick Left	Down	Up	Serial Data In	Serial Clock In	37151

6522 VIA 2

Hex									Decimal
9120	Joystick Right				Tape Out				37152
	Keyboard Row Select								
9121	Keyboard Column Input								37153
9122	Data Direction Register B (for $9120)								37154
9123	Data Direction Register A (for $9121)								37155
9124	T1-L	Cassette Tape Read;							37156
9125	T1-H	Keyboard and Clock							37157
9126	T1-Latch L	Interrupt Timing							37158
9127	T1 Latch H								37159
9128	T2-L	Serial Bus Timing							37160
9129	T2-H	Tape R/W Timing							37161
912A	Shift Register (* unused)								37162
912B	T1 Control		T2 Ctrl	Shift Register Control			PB LE	PA LE	37163
912C	Serial Bus Data Out			CB1 Ctrl	Serial Clock Line Out			CA1 Ctrl	37164
912D	IRQ:	T1	T2	CB1 SRQ In			CA1 Tape In		37165
912E	IRQ En.	T1 Enab	T2 Enab	CB1 En.			CA1 En.		37166
912F	*Unused (see $9121)								37167

SuperChart: VIC 20 / Commodore 64

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
0	00		@	end-line	BRK	0
1	01		A		ORA(I,X)	1
2	02		B			2
3	03	stop	C			3
4	04		D			4
5	05	white	E		ORA Z	5
6	06		F		ASL Z	6
7	07		G			7
8	08	lock	H		PHP	8
9	09	unlock	I		ORA #	9
10	0A		J		ASL A	10
11	0B		K			11
12	0C		L			12
13	0D	car ret	M		ORA	13
14	0E	text	N		ASL	14
15	0F		O			15
16	10		P		BPL	16
17	11	cur down	Q		ORA(I),Y	17
18	12	reverse	R			18
19	13	cur home	S			19
20	14	delete	T			20
21	15		U		ORA Z,X	21
22	16		V		ASL Z,X	22
23	17		W			23
24	18		X		CLC	24
25	19		Y		ORA Y	25
26	1A		Z			26
27	1B		[			27
28	1C	red	\			28
29	1D	cur right	]		ORA X	29
30	1E	green	↑		ASL X	30
31	1F	blue	←			31
32	20	space	space	space	JSR	32
33	21	!	!	!	AND(I,X)	33
34	22	"	"	"		34
35	23	#	#	#		35
36	24	$	$	$	BIT Z	36
37	25	%	%	%	AND Z	37
38	26	&	&	&	ROL Z	38
39	27	'	'	'		39
40	28	(	(	(	PLP	40
41	29	)	)	)	AND #	41
42	2A	*	*	*	ROL A	42
43	2B	+	+	+		43
44	2C	,	,	,	BIT	44
45	2D	–	–	–	AND	45
46	2E	.	.	.	ROL	46
47	2F	/	/	/		47
48	30	0	0	0	BMI	48
49	31	1	1	1	AND(I),Y	49
50	32	2	2	2		50
51	33	3	3	3		51
52	34	4	4	4		52
53	35	5	5	5	AND Z,X	53
54	36	6	6	6	ROL Z,X	54
55	37	7	7	7		55
56	38	8	8	8	SEC	56
57	39	9	9	9	AND Y	57
58	3A	:	:	:		58
59	3B	;	;	;		59
60	3C	<	<	<		60
61	3D	=	=	=	AND X	61
62	3E	>	>	>	ROL X	62
63	3F	?	?	?		63

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
64	40	@	[graphic]	@	RTI	64
65	41	A	[graphic],a	A	EOR(I,X)	65
66	42	B	[graphic],b	B		66
67	43	C	[graphic],c	C		67
68	44	D	[graphic],d	D		68
69	45	E	[graphic],e	E	EOR Z	69
70	46	F	[graphic],f	F	LSR Z	70
71	47	G	[graphic],g	G		71
72	48	H	[graphic],h	H	PHA	72
73	49	I	[graphic],i	I	EOR #	73
74	4A	J	[graphic],j	J	LSR A	74
75	4B	K	[graphic],k	K		75
76	4C	L	[graphic],l	L	JMP	76
77	4D	M	[graphic],m	M	EOR	77
78	4E	N	[graphic],n	N	LSR	78
79	4F	O	[graphic],o	O		79
80	50	P	[graphic],p	P	BVC	80
81	51	Q	[graphic],q	Q	EOR(I),Y	81
82	52	R	[graphic],r	R		82
83	53	S	[graphic],s	S		83
84	54	T	[graphic],t	T		84
85	55	U	[graphic],u	U	EOR Z,X	85
86	56	V	[graphic],v	V	LSR Z,X	86
87	57	W	[graphic],w	W		87
88	58	X	[graphic],x	X	CLI	88
89	59	Y	[graphic],y	Y	EOR Y	89
90	5A	Z	[graphic],z	Z		90
91	5B	[	[graphic]	[		91
92	5C	£	[graphic]	£		92
93	5D	]	[graphic]	]	EOR X	93
94	5E	↑	[graphic],[graphic]	↑	LSR X	94
95	5F	←	[graphic],[graphic]	←		95
96	60		[graphic]		RTS	96
97	61		[graphic]		ADC(I,X)	97
98	62		[graphic]			98
99	63		[graphic]			99
100	64		[graphic]			100
101	65		[graphic]		ADC Z	101
102	66		[graphic]		ROR Z	102
103	67		[graphic]			103
104	68		[graphic]		PLA	104
105	69		[graphic],[graphic]		ADC #	105
106	6A		[graphic]		ROR A	106
107	6B		[graphic]			107
108	6C		[graphic]		JMP(I)	108
109	6D		[graphic]		ADC	109
110	6E		[graphic]		ROR	110
111	6F		[graphic]			111
112	70		[graphic]		BVS	112
113	71		[graphic]		ADC(I),Y	113
114	72		[graphic]			114
115	73		[graphic]			115
116	74		[graphic]			116
117	75		[graphic]		ADC Z,X	117
118	76		[graphic]		ROR Z,X	118
119	77		[graphic]			119
120	78		[graphic]		SEI	120
121	79		[graphic]		ADC Y	121
122	7A		[graphic],[graphic]			122
123	7B		[graphic]			123
124	7C		[graphic]			124
125	7D		[graphic]		ADC X	125
126	7E		[graphic]		ROR X	126
127	7F		[graphic]			127

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
128	80		@	END		128
129	81	orange	A	FOR	STA(I,X)	129
130	82		B	NEXT		130
131	83	load & run	C	DATA		131
132	84		D	INPUT#	STY Z	132
133	85	F1	E	INPUT	STA Z	133
134	86	F3	F	DIM	STX Z	134
135	87	F5	G	READ		135
136	88	F7	H	LET	DEY	136
137	89	F2	I	GOTO		137
138	8A	F4	J	RUN	TXA	138
139	8B	F6	K	IF		139
140	8C	F8	L	RESTORE	STY	140
141	8D	car ret	M	GOSUB	STA	141
142	8E	graphics	N	RETURN	STX	142
143	8F		O	REM		143
144	90	black	P	STOP	BCC	144
145	91	cur up	Q	ON	STA(I),Y	145
146	92	rvs off	R	WAIT		146
147	93	clear	S	LOAD		147
148	94	insert	T	SAVE	STY Z,X	148
149	95	brown	U	VERIFY	STA Z,X	149
150	96	lt. red	V	DEF	STX Z,Y	150
151	97	dk. grey	W	POKE		151
152	98	md. grey	X	PRINT#	TYA	152
153	99	lt. green	Y	PRINT	STA Y	153
154	9A	lt. blue	Z	CONT	TXS	154
155	9B	lt. grey	[	LIST		155
156	9C	magenta	£	CLR		156
157	9D	cur left	]	CMD	STA X	157
158	9E	yellow	↑	SYS		158
159	9F	cyan	←	OPEN		159
160	A0			CLOSE	LDY #	160
161	A1		!	GET	LDA(I,X)	161
162	A2		"	NEW	LDX #	162
163	A3		#	TAB(		163
164	A4		$	TO	LDY Z	164
165	A5		%	FN	LDA Z	165
166	A6		&	SPC(	LDX Z	166
167	A7		'	THEN		167
168	A8		(	NOT	TAY	168
169	A9		)	STEP	LDA #	169
170	AA		*	+	TAX	170
171	AB		+	−		171
172	AC		,	*	LDY	172
173	AD		-	/	LDA	173
174	AE		.	↑	LDX	174
175	AF		/	AND		175
176	B0		0	OR	BCS	176
177	B1		1	>	LDA(I),Y	177
178	B2		2	=		178
179	B3		3	<		179
180	B4		4	SGN	LDY Z,X	180
181	B5		5	INT	LDA Z,X	181
182	B6		6	ABS	LDX Z,Y	182
183	B7		7	USR		183
184	B8		8	FRE	CLV	184
185	B9		9	POS	LDA Y	185
186	BA		:	SQR	TSX	186
187	BB		;	RND		187
188	BC		<	LOG	LDY X	188
189	BD		=	EXP	LDA X	189
190	BE		>	COS	LDX Y	190
191	BF		?	SIN		191

DECIMAL	HEX	ASCII	SCREEN	BASIC	6502	DECIMAL
192	C0		Reverse of ASCII	TAN	CPY #	192
193	C1	,a		ATN	CMP(I),X	193
194	C2	,b		PEEK		194
195	C3	,c		LEN		195
196	C4	,d		STR$	CPY Z	196
197	C5	,e		VAL	CMP Z	197
198	C6	,f		ASC	DEC Z	198
199	C7	,g		CHR$		199
200	C8	,h		LEFT$	INY	200
201	C9	,i		RIGHT$	CMP #	201
202	CA	,j		MID$	DEX	202
203	CB	,k		GO		203
204	CC	,l			CPY	204
205	CD	,m			CMP	205
206	CE	,n			DEC	206
207	CF	,o				207
208	D0	,p			BNE	208
209	D1	,q			CMP(I),Y	209
210	D2	,r				210
211	D3	,s				211
212	D4	,t				212
213	D5	,u			CMP Z,X	213
214	D6	v			DEC Z,X	214
215	D7	,w				215
216	D8	,x			CLD	216
217	D9	,y			CMP Y	217
218	DA	,z				218
219	DB					219
220	DC					220
221	DD				CMP X	221
222	DE				DEC X	222
223	DF					223
224	E0				CPX #	224
225	E1				SBC(I),X	225
226	E2					226
227	E3					227
228	E4				CPX Z	228
229	E5				SBC Z	229
230	E6				INC Z	230
231	E7					231
232	E8				INX	232
233	E9				SBC #	233
234	EA				NOP	234
235	EB					235
236	EC				CPX	236
237	ED				SBC	237
238	EE				INC	238
239	EF					239
240	F0				BEQ	240
241	F1				SBC(I),Y	241
242	F2					242
243	F3					243
244	F4					244
245	F5				SBC Z,X	245
246	F6				INC Z,X	246
247	F7					247
248	F8				SED	248
249	F9				SBC Y	249
250	FA					250
251	FB					251
252	FC					252
253	FD				SBC X	253
254	FE				INC X	254
255	FF			π		255

Commodore 64 Memory Map

0000	0	Chip directional register
0001	1	Chip I/O; memory & tape control
0003 - 0004	3-4	Float-Fixed vector
0005 - 0006	5-6	Fixed-Float vector
0007	7	Search character
0008	8	Scan-quotes flag
0009	9	TAB column save
000A	10	0 = LOAD, 1 = VERIFY
000B	11	Input buffer pointer/# subscrpt
000C	12	Default DIM flag
000D	13	Type: FF = string, 00 = numeric
000E	14	Type: 80 = integer, 00 = floating point
000F	15	DATA scan/LIST quote/memry flag
0010	16	Subscript/FNx flag
0011	17	0 = INPUT;$40 = GET;$98 = READ
0012	18	ATN sign/Comparison eval flag
0013	19	Current I/O prompt flag
0014 - 0015	20-21	Integer value
0016	22	**Pointer:** temporary string stack
0017 - 0018	23-24	Last temp string vector
0019 - 0021	25-33	Stack for temporary strings
0022 - 0025	34-37	Utility pointer area
0026 - 002A	38-42	Product area for multiplication
002B - 002C	43-44	**Pointer:** Start of BASIC
002D - 002E	45-46	**Pointer:** Start of Variables
002F - 0030	47-48	**Pointer:** Start of Arrays
0031 - 0032	49-50	**Pointer:** End of Arrays
0033 - 0034	51-52	**Pointer:** String Storage (moving down)
0035 - 0036	53-54	**Pointer:** Utility String
0037 - 0038	55-56	**Pointer:** Limit of Memory
0039 - 003A	57-58	Current BASIC line number
003B - 003C	59-60	Previous BASIC line number
003D - 003E	61-62	**Pointer:** BASIC statement for CONT
003F - 0040	63-64	Current DATA line number
0041 - 0042	65-66	Current DATA address
0043 - 0044	67-68	Input vector
0045 - 0046	69-70	Current variable name
0047 - 0048	71-72	Current variable address
0049 - 004A	73-74	Variable pointer for FOR/NEXT
004B - 004C	75-76	Y-save; op-save; BASIC pointer save
004D	77	Comparison symbol accumulator
004E - 0053	78-83	Misc work area, pointers, etc
0054 - 0056	84-86	Jump vector for functions
0057 - 0060	87-96	Misc numeric work area
0061	97	Accum#1: Exponent
0062 - 0065	98-101	Accum#1: Mantissa
0066	102	Accum#1: Sign
0067	103	Series evaluation constant pointer
0068	104	Accum#1 hi-order (overflow)
0069	105	Accum#2: Exponent, etc.
006A - 006D	106-109	Accum#2: Mantissa
006E	110	Accum#2: Exponent, etc.
006F	111	Sign comparison, Acc#1 vs #2
0070	112	Accum#1 lo-order (rounding)
0071 - 0072	113-114	Cassette buff len/Series pointer
0073 - 008A	115-138	CHRGET subroutine; get BASIC char
007A - 007B	122-123	BASIC pointer (within subrtn)
008B - 008F	139-143	RND seed value
0090	144	Status word ST
0091	145	Keyswitch PIA: STOP and RVS flags
0092	146	Timing constant for tape
0093	147	Load = 0, Verify = 1
0094	148	Serial output: deferred char flag
0095	149	Serial deferred character
0096	150	Tape EOT received
0097	151	Register save
0098	152	How many open files
0099	153	Input device, normally 0
009A	154	Output CMD device, normally 3
009B	155	Tape character parity
009C	156	Byte-received flag
009D	157	Direct = $80/RUN = 0 output control
009E	158	Tp Pass 1 error log/char buffer
009F	159	Tp Pass 2 err log corrected
00A0 - 00A2	160-162	Jiffy Clock HML
00A3	163	Serial bit count/EOI flag
00A4	164	Cycle count
00A5	165	Countdown,tape write/bit count
00A6	166	Tape buffer pointer
00A7	167	Tp Wrt ldr count/Rd pass/inbit
00A8	168	Tp Wrt new byte/Rd error/inbit cnt
00A9	169	Wrt start bit/Rd bit err/stbit
00AA	170	Tp Scan;Cnt;Ld;End/byte assy
00AB	171	Wr lead length/Rd checksum/parity
00AC - 00AD	172-173	**Pointer:** tape bufr, scrolling
00AE - 00AF	174-175	Tape end adds/End of program
00B0 - 00B1	176-177	Tape timing constants
00B2 - 00B3	178-179	**Pointer:** Start of Tape Buffer
00B4	180	1 = Tp timer enabled; bit count
00B5	181	Tp EOT/RS232 next bit to send
00B6	182	Read character error/outbyte buf
00B7	183	# characters in file name
00B8	184	Current logical file
00B9	185	Current secndy address
00BA	186	Current device
00BB - 00BC	187-188	Pointer to file name
00BD	189	Wr shift word/Rd input char
00BE	190	# blocks remaining to Wr/Rd
00BF	191	Serial word buffer
00C0	192	Tape motor interlock
00C1 - 00C2	193-194	I/O start address
00C3 - 00C4	195-196	Kernel setup pointer
00C5	197	Last key pressed
00C6	198	# chars in keybd buffer
00C7	199	Screen reverse flag
00C8	200	End-of-line for input pointer
00C9 - 00CA	201-202	Input cursor log (row, column)
00CB	203	Which key: 64 if no key
00CC	204	0 = flash cursor
00CD	205	Cursor timing countdown
00CE	206	Character under cursor
00CF	207	Cursor in blink phase
00D0	208	Input from screen/from keyboard
00D1 - 00D2	209-210	Pointer to screen line
00D3	211	Position of cursor on above line
00D4	212	0 = direct cursor, else programmed
00D5	213	Current screen line length
00D6	214	Row where curosr lives
00D7	215	Last inkey/checksum/buffer
00D8	216	# of INSERTs outstanding
00D9 - 00F2	217-242	Screen line link table
00F3 - 00F4	243-244	Screen colour pointer
00F5 - 00F6	245-246	Keyboard pointer
00F7 - 00F8	247-248	RS-232 Rcv pntr
00F9 - 00FA	249-250	RS-232 Tx pntr
00FF - 010A	256-266	Floating to ASCII work area
0100 - 013E	256-318	Tape error log
0100 - 01FF	256-511	Processor stack area
0200 - 0258	512-600	BASIC input buffer
0259 - 0262	601-610	Logical file table
0263 - 026C	611-620	Device # table
026D - 0276	621-630	Sec Adds table
0277 - 0280	631-640	Keyboard buffer
0281 - 0282	641-642	Start of BASIC Memory
0283 - 0284	643-644	Top of BASIC Memory
0285	645	Serial bus timeout flag
0286	646	Current colour code
0287	647	Colour under cursor
0288	648	Screen memory page
0289	649	Max size of keybd buffer
028A	650	Repeat all keys
028B	651	Repeat speed counter
028C	652	Repeat delay counter
028D	653	Keyboard Shift/Control flag
028E	654	Last shift pattern
028F - 0290	655-656	Keyboard table setup pointer
0291	657	Keyboard shift mode
0292	658	0 = scroll enable
0293	659	RS-232 control reg
0294	660	RS-232 command reg
0295 - 0296	661-662	Bit timing
0297	663	RS-232 status
0298	664	# bits to send
0299 - 029A	665-666	RS-232 speed/code
029B	667	RS232 receive pointer
029C	668	RS232 input pointer
029D	669	RS232 transmit pointer
029E	670	RS232 output pointer
029F - 02A0	671-672	IRQ save during tape I/O
02A1	673	CIA 2 (NMI) Interrupt Control
02A2	674	CIA 1 Timer A control log
02A3	675	CIA 1 Interrupt Log
02A4	676	CIA 1 Timer A enabled flag
02A5	677	Screen row marker
02C0 - 02FE	704-766	(Sprite 11)
0300 - 0301	768-769	Error message link
0302 - 0303	770-771	BASIC warm start link
0304 - 0305	772-773	Crunch BASIC tokens link
0306 - 0307	774-775	Print tokens link
0308 - 0309	776-777	Start new BASIC code link
030A - 030B	778-779	Get arithmetic element link
030C	780	SYS A-reg save
030D	781	SYS X-reg save
030E	782	SYS Y-reg save
030F	783	SYS status reg save
0310 - 0312	784-786	USR function jump JMP B248
0314 - 0315	788-789	Hardware interrupt vector (EA31)
0316 - 0317	790-791	Break interrupt vector (FE66)
0318 - 0319	792-793	NMI interrupt vector (FE47)
031A - 031B	794-795	OPEN vector (F34A)
031C - 031D	796-797	CLOSE vector (F291)
031E - 031F	798-799	Set-input vector (F20E)
0320 - 0321	800-801	Set-output vector (F250)
0322 - 0323	802-803	Restore I/O vector (F333)
0324 - 0325	804-805	INPUT vector (F157)
0326 - 0327	806-807	Output vector (F1CA)
0328 - 0329	808-809	Test-STOP vector (F6ED)
032A - 032B	810-811	GET vector (F13E)
032C - 032D	812-813	Abort I/O vector (F32F)
032E - 032F	814-815	Warm start vector (FE66)
0330 - 0331	816-817	LOAD link (F4A5)
0332 - 0333	818-819	SAVE link (F5ED)
033C - 03FB	828-1019	Cassette buffer
0340 - 037E	832-894	(Sprite 13)
0380 - 03BE	896-958	(Sprite 14)
03C0 - 03FE	960-1022	(Sprite 15)
0400 - 07F7	1024-2039	Screen memory (default)
07F8 - 07FF	2040-2047	Sprite Pointers (default)
0800 - 9FFF	2048-40959	BASIC ROM memory
8000 - 9FFF	32768-40959	Alternate: ROM plug-in area
A000 - BFFF	40960-49151	ROM: BASIC
A000 - BFFF	40960-59151	Alternate: RAM
C000 - CFFF	49152-53247	RAM memory, including alternate
D000 - D02E	53248-53294	Video Chip (6566)
D400 - D41C	54272-54300	Sound Chip (6581 SID)
D800 - DBFF	55296-56319	Colour nybble memory
DC00 - DC0F	56320-56335	Interface chip 1, IRQ (6526 CIA)
DD00 - DD0F	56576-56591	Interface chip 2, NMI (6526 CIA)
D000 - DFFF	53248-53294	Alternate: Character set
E000 - FFFF	57344-65535	ROM: Operating System
E000 - FFFF	57344-65535	Alternate: RAM
FF81 - FFF5	65409-65525	Jump Table, Including:
FFC6		Set Input channel
FFC9		Set Output channel
FFCC		Restore default I/O channels
FFCF		INPUT
FFD2		PRINT
FFE1		Test Stop key
FFE4		GET

Commodore 64 ROM Routines

A000	ROM control vectors
A00C	Keyword action vectors
A052	Function vectors
A080	Operator vectors
A09E	Keywords
A19E	Error messages
A328	Error message vectors
A365	Misc messages
A38A	Scan stack for FOR/GOSUB
A3B8	Move memory
A3FB	Check stack depth
A408	Check memory space
A435	'out of memory'
A437	Error routine
A469	BREAK entry
A474	'ready.'
A480	Ready for BASIC
A49C	Handle new line
A533	Re-chain lines
A560	Receive input line
A579	Crunch tokens
A613	Find BASIC line
A642	Perform [NEW]
A65E	Perform [CLR]
A68E	Back up text pointer
A69C	Perform [LIST]
A742	Perform [FOR]
A7ED	Execute statement
A81D	Perform [RESTORE]
A82C	Break
A82F	Perform [STOP]
A831	Perform [END]
A857	Perform [CONT]
A871	Perform [RUN]
A883	Perform [GOSUB]
A8A0	Perform [GOTO]
A8D2	Perform [RETURN]
A8F8	Perform [DATA]
A906	Scan for next statement
A928	Perform [IF]
A93B	Perform [REM]
A94B	Perform [ON]
A96B	Get fixed point number
A9A5	Perform [LET]
AA80	Perform [PRINT#]
AA86	Perform [CMD]
AAA0	Perform [PRINT]
AB1E	Print string from (y.a)
AB3B	Print format character
AB4D	Bad input routine
AB7B	Perform [GET]
ABA5	Perform [INPUT#]
ABBF	Perform [INPUT]
ABF9	Prompt & input
AC06	Perform [READ]
ACFC	Input error messages
AD1E	Perform [NEXT]
AD78	Type match check
AD9E	Evaluate expression
AEA8	Constant - PI
AEF1	Evaluate within brackets
AEF7	')'
AEFF	comma..
AF08	Syntax error
AF14	Check range
AF28	Search for variable
AFA7	Setup FN reference
AFE6	Perform [OR]
AFE9	Perform [AND]
B016	Compare
B081	Perform [DIM]
B08B	Locate variable
B113	Check alphabetic
B11D	Create variable
B194	Array pointer subroutine
B1A5	Value 32768
B1B2	Float-fixed conversion
B1D1	Set up array
B245	'BAD SUBSCRIPT'
B248	'ILLEGAL QUANTITY'
B34C	Compute array size
B37D	Perform [FRE]
B391	Fix-float conversion
B39E	Perform [POS]
B3A6	Check direct
B3B3	Perform [DEF]
B3E1	Check FN syntax
B3F4	Perform [FN]
B465	Perform [STR$]
B475	Calculate string vector
B487	Set up string
B4F4	Make room for string
B526	Garbage collection
B5BD	Check salvageability
B606	Collect string
B63D	Concatenate
B67A	Build string to memory
B6A3	Discard unwanted string
B6DB	Clean descriptor stack
B6EC	Perform [CHR$]
B700	Perform [LEFT$]
B72C	Perform [RIGHT$]
B737	Perform [MID$]
B761	Pull string parameters
B77C	Perform [LEN]
B782	Exit string-mode
B78B	Perform [ASC]
B79B	Input byte parameter
B7AD	Perform [VAL]
B7EB	Parameters for POKE/WAIT
B7F7	Float-fixed
B80D	Perform [PEEK]
B824	Perform [POKE]
B82D	Perform [WAIT]
B849	Add 0.5
B850	Subtract-from
B853	Perform [subtract]
B86A	Perform [add]
B947	Complement FAC#1
B97E	'overflow'
B983	Multiply by zero byte
B9EA	Perform [LOG]
BA2B	Perform [multiply]
BA59	Multiply-a-bit
BA8C	Memory to FAC#2
BAB7	Adjust FAC#1/#2
BAD4	Underflow/overflow
BAE2	Multiply by 10
BAF9	+ 10 in floating pt
BAFE	Divide by 10
BB12	Perform [divide]
BBA2	Memory to FAC#1
BBC7	FAC#1 to memory
BBFC	FAC#2 to FAC#1
BC0C	FAC#1 to FAC#2
BC1B	Round FAC#1
BC2B	Get sign
BC39	Perform [SGN]
BC58	Perform [ABS]
BC5B	Compare FAC#1 to mem
BC9B	Float-fixed
BCCC	Perform [INT]
BCF3	String to FAC
BD7E	Get ASCII digit
BDC2	Print 'IN..'
BDCD	Print line number
BDDD	Float to ASCII
BF16	Decimal constants
BF3A	TI constants
BF71	Perform [SQR]
BF7B	Perform [power]
BFB4	Perform [negative]
BFED	Perform [EXP]
E043	Series eval 1
E059	Series eval 2
E097	Perform [RND]
E0F9	?? breakpoints ??
E12A	Perform [SYS]
E156	Perform [SAVE]
E165	Perform [VERIFY]
E168	Perform [LOAD]
E1BE	Perform [OPEN]
E1C7	Perform [CLOSE]
E1D4	Parameters for LOAD/SAVE
E206	Check default parameters
E20E	Check for comma
E219	Parameters for open/close
E264	Perform [COS]
E26B	Perform [SIN]
E2B4	Perform [TAN]
E30E	Perform [ATN]
E37B	Warm restart
E394	Initialize
E3A2	CHRGET for zero page
E3BF	Initialize BASIC
E447	Vectors for $300
E453	Initialize vectors
E45F	Power-up message
E500	Get I/O address
E505	Get screen size
E50A	Put/get row/column
E518	Initialize I/O
E544	Clear screen
E566	Home cursor
E56C	Set screen pointers
E5A0	Set I/O defaults
E5B4	Input from keyboard
E632	Input from screen
E684	Quote test
E691	Setup screen print
E6B6	Advance cursor
E6ED	Retreat cursor
E701	Back into previous line
E716	Output to screen
E87C	Go to next line
E891	Perform <return>
E8A1	Check line decrement
E8B3	Check line increment
E8CB	Set colour code
E8DA	Colour code table
E8EA	Scroll screen
E965	Open space on screen
E9C8	Move a screen line
E9E0	Synchronize colour transfer
E9F0	Set start-of-line
E9FF	Clear screen line
EA13	Print to screen
EA24	Synchronize colour pointer
EA31	Interrupt - clock etc
EA87	Read keyboard
EB79	Keyboard select vectors
EB81	Keyboard 1 - unshifted
EBC2	Keyboard 2 - shifted
EC03	Keyboard 3 - 'comm'
EC44	Graphics/text contrl
EC4F	Set graphics/text mode
EC78	Keyboard 4
ECB9	Video chip setup
ECE7	Shift/run equivalent
ECF0	Screen In address low
ED09	Send 'talk'
ED0C	Send 'listen'
ED40	Send to serial bus
EDB2	Serial timeout
EDB9	Send listen SA
EDBE	Clear ATN
EDC7	Send talk SA
EDCC	Wait for clock
EDDD	Send serial deferred
EDEF	Send 'untalk'
EDFE	Send 'unlisten'
EE13	Receive from serial bus
EE85	Serial clock on
EE8E	Serial clock off
EE97	Serial output '1'
EEA0	Serial output '0'
EEA9	Get serial in & clock
EEB3	Delay 1 ms
EEBB	RS-232 send
EF06	Send new RS-232 byte
EF2E	No-DSR error
EF31	No-CTS error
EF3B	Disable timer
EF4A	Compute bit count
EF59	RS232 receive
EF7E	Setup to receive
EFC5	Receive parity error
EFCA	Receive overflow
EFCD	Receive break
EFD0	Framing error
EFE1	Submit to RS232
F00D	No-DSR error
F017	Send to RS232 buffer
F04D	Input from RS232
F086	Get from RS232
F0A4	Check serial bus idle
F0BD	Messages
F12B	Print if direct
F13E	Get..
F14E	..from RS232
F157	Input
F199	Get.. tape/serial/rs232
F1CA	Output..
F1DD	..to tape
F20E	Set input device
F250	Set output device
F291	Close file
F30F	Find file
F31F	Set file values
F32F	Abort all files
F333	Restore default I/O
F34A	Do file open
F3D5	Send SA
F409	Open RS232
F49E	LOAD program
F5AF	'searching'
F5C1	Print filename
F5D2	'loading/verifying'
F5DD	SAVE program
F68F	Print 'SAVING'
F69B	Bump clock
F6BC	Log PIA key reading
F6DD	Get time
F6E4	Set time
F6ED	Check stop key
F6FB	Output error messages
F72D	Find any tape headr
F76A	Write tape header
F7D0	Get buffer address
F7D7	Set buffer start/end pointers
F7EA	Find specific header
F80D	Bump tape pointer
F817	'press play..'
F82E	Check tape status
F838	'press record..'
F841	Initiate tape read
F864	Initiate tape write
F875	Common tape code
F8D0	Check tape stop
F8E2	Set read timing
F92C	Read tape bits
FA60	Store tape chars
FB8E	Reset pointer
FB97	New character setup
FBA6	Send transition to tape
FBC8	Write data to tape
FBCD	IRQ entry point
FC57	Write tape leader
FC93	Restore normal IRQ
FCB8	Set IRQ vector
FCCA	Kill tape motor
FCD1	Check r/w pointer
FCDB	Bump r/w pointer
FCE2	Power reset entry
FD02	Check 8-rom
FD10	8-ROM mask
FD15	Kernal reset
FD1A	Kernal move
FD30	Vectors
FD50	Initialize system constnts
FD9B	IRQ vectors
FDA3	Initialize I/O
FDDD	Enable timer
FDF9	Save filename data
FE00	Save file details
FE07	Get status
FE18	Flag status
FE1C	Set status
FE21	Set timeout
FE25	Read/set top of memory
FE27	Read top of memory
FE2D	Set top of memory
FE34	Read/set bottom of memory
FE43	NMI entry
FE66	Warm start
FEB6	Reset IRQ & exit
FEBC	Interrupt exit
FEC2	RS-232 timing table
FED6	NMI RS-232 in
FF07	NMI RS-232 out
FF43	Fake IRQ
FF48	IRQ entry
FF81	Jumbo jump table
FFFA	Hardware vectors

6566 Video Chip
C64 Control & Miscellaneous Registers

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Decimal
D011		Extended Clr. Mode	Bit Map	Display Enable	Row Select	Y-Scroll			53265
D012	Raster Register								53266
D013	Light Pen Input X								53267
D014	Light Pen Input Y								53268
D016	x	x	Reset	Multi Colour	Column Select	X-Scroll			53270
D018	Screen VM13	VM12	VM11	VM10	Character Base CB13	CB12	CB11	x	53272
D019	IRQ	Interrupt Sense:			Light Pen	Spr-Spr Collision	Spr-Back Collision	Raster	53273
D01A		Interrupt Enable:			Light Pen	Spr-Spr Collisions	Spr-Back Collisions	Raster	53274

Colour Registers

Address	Bits 7–4	Bits 3–0	Decimal
D020	X	Exterior Colour (Border)	53280
D021	X	Background Colour #0	53281
D022	X	Background Colour #1	53282
D023	X	Background Colour #2	53283
D024	X	Background Colour #3	53284
D025	X	Sprite MultiColour #0	53285
D026	X	Sprite MultiColour #1	53286

6566 Video Chip
C64 Sprite Registers

Sprite 0	Sprite 7	Register	Sprite 0	Sprite 7
D000	D00E	X Position	53248	53262
D001	D00F	Y Position	53249	53263
D027	D02E	Sprite Colour	53287	53294

Bit For Sprite#: 7 6 5 4 3 2 1 0

Address	Register	Decimal
D010	X-Position High	53264
D015	Sprite Enable Flags	53269
D017	Y-Expand	53271
D01B	Background Priority	53275
D01C	Sprite MultiColour Mode	53276
D01D	X-Expand	53277
D01E	Interrupt: Sprite Collision	53278
D01F	Interrupt: Background Collision	53279

CIA 1 (IRQ) (6526)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Name	Decimal
$DC00	Paddle Sel A	B		Joystick 0 Fire	Right	Left	Down*	Up	PRA	56320
	Keyboard Row Select (inverted)									
$DC01				Joystick 1 Fire	Right	Left	Down	Up	PRB	56321
	Keyboard Column Read									
$DC02	$FF – All Output								DDRA	56322
$DC03	$00 – All Input								DDRB	56323
$DC04	Timer A								TAL	56324
$DC05									TAH	56325
$DC06	Timer B								TBL	56326
$DC07									TBH	56327
$DC0D				Tape Input			Timer Interrupt B	A	ICR	56333
$DC0E					One Shot	Out Mode	Time PB6 Out	Timer A Start	CRA	56334
$DC0F					One Shot	Out Mode	Time PB7 Out	Timer B Start	CRB	56335

CIA 2 (NMI) (6526)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Name	Decimal
$DD00	Serial IN	Clock IN	Serial OUT	Clock OUT	ATN OUT	RS-232 OUT	VIC II addr 15	VIC II addr 14	PRA	56576
$DD01	DSR IN	CTS IN		DCD* IN	RI* IN	DTR OUT	RTS OUT	RS-232 IN	PRB	56577
$DD02	$3F – Serial								DDRA	56578
$DD03	$00 – P.U.P. All Input or $06 – RS-232								DDRB	56579
$DD04	Timer A								TAL	56580
$DD05									TAH	56581
$DD06	Timer B								TBL	56582
$DD07									TBH	56583
$DD0D				RS-232 IN			Timer Interrupt B	A	ICR	56589
$DD0E								Timer A Start	CRA	56590
$DD0F								Timer B Start	CRB	56591

* Connected but not used by O.S.

Processor I/O Port (6510)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Name	
$0000	IN	IN	OUT	IN	OUT	OUT	OUT	OUT	DDR	0
$0001			Tape Motor	Tape Sense	Tape Write	D-ROM Switch	EF RAM Switch	AB RAM Switch	PR	1

SID (6581)

Voice 1	Voice 2	Voice 3	Register	Voice 1	Voice 2	Voice 3
$D400	$D407	$D40E	Frequency L	54272	54279	54286
$D401	$D408	$D40F	Frequency H	54273	54280	54287
$D402	$D409	$D410	Pulse Width L	54274	54281	54288
$D403	$D40A	$D411	0 0 0 0 / Pulse Width H	54275	54282	54289
$D404	$D40B	$D412	Voice Type: NSE, PUL, SAW, TRI; Key	54276	54283	54290
$D405	$D40C	$D413	Attack Time 2ms – 8ms; Decay Time 6ms – 24 sec	54277	54284	54291
$D406	$D40D	$D414	Sustain Level; Release Time 6ms 24 sec	54278	54285	54292

Voices (write only)

Address	Register	Decimal
$D415	0 0 0 0 0 / Filter Frequency L	54293
$D416	Filter Frequency H	54294
$D417	Resonance; Filter Voices: Ext, V3, V2, V1	54295
$D418	V3 off; Passband: HI, BP, LO; Master Volume	54296

Filter & Volume (write only)

Address	Register	Decimal
$D419	Paddle X (A/D #1)	54297
$D41A	Paddle Y (A/D #2)	54298
$D41B	Noise 3 (random)	54299
$D41C	Envelope 3	54300

Sense (read only)

Note: Special Voice Features
(TEST, RING MOD, SYNC)
are omitted from the above diagram.

VIC 20 / Commodore 64 Memory Map

With Zero Page Contents at Power-Up

There are some differences between the 20 and 64 as indicated.

Location				Contents				Description
Hex		Dec		VIC Hex	VIC Dec	C64 Hex	C64 Dec	
00 -02	00	0–2	0	4C	76	2F	47	USR Jump. 64: Chip directional reg.
	01		1	48	72	37	55	64: Chip I/O; memory & tape control
	02		2	D2	210	33	51	20: JMP $D248. 64: Unused
03 -04	03	3–4	3	AA	170	AA	170	Float–Fixed vector
	04		4	D1	209	B1	177	
05 -06	05	5–6	5	91	145	91	145	Fixed–Float vector
	06		6	D3	211	B3	179	
07	07	7	7	22	34	22	34	Search character
08	08	8	8	22	34	22	34	Scan–quotes flag
09	09	9	9	00	0	00	0	TAB column save
0A	0A	10	10	00	0	00	0	0 = LOAD, 1 = VERIFY
0B	0B	11	11	4C	76	4C	76	Input buffer pointer/# subscripts
0C	0C	12	12	00	0	00	0	Default DIM flag
0D	0D	13	13	00	0	00	0	Type: FF = string, 00 = numeric
0E	0E	14	14	00	0	00	0	Type: 80 = integer, 00 = floating pt
0F	0F	15	15	00	0	00	0	DATA scan/LIST quote/memory flag
10	10	16	16	00	0	00	0	Subscript/FNx flag
11	11	17	17	00	0	00	0	0 = INPUT;$40 = GET;$98 = READ
12	12	18	18	00	0	00	0	ATN sign/Comparison eval. flag
13	13	19	19	05	5	05	5	Current I/O prompt flag
14 -15	14	20–21	20	14	20	14	20	Integer value
	15		21	00	0	00	0	
16	16	22	22	19	25	19	25	**Pointer:** Temporary string stack
17 -18	17	23–24	23	16	22	16	22	Last temp string vector
	18		24	00	0	00	0	
19 -21	19	25–33	25	02	25	02	2	Stack for temporary strings
	1A		26	FE	254	FE	254	
	1B		27	1D	29	9F	159	
	1C		28	0	0	00	0	
	1D		29	00	0	00	0	
	1E		30	00	0	00	0	
	1F		31	00	0	1E	30	
	20		32	00	0	00	0	
	21		33	00	0	00	0	
22 -25	22	34–37	34	05	5	05	5	Utility pointer area
	23		35	10	16	08	8	
	24		36	F3	243	F3	243	
	25		37	01	1	01	1	
26 -2A	26	38–42	38	00	0	00	0	Product area for multiplication
	27		39	00	0	00	0	
	28		40	00	0	00	0	
	29		41	00	0	00	0	
	2A		42	00	0	00	0	
2B -2C	2B	43–44	43	01	1	01	1	**Pointer:** Start of BASIC
	2C		44	10	16	08	8	
2D -2E	2D	45–46	45	03	3	03	3	**Pointer:** Start of Variables
	2E		46	10	16	08	8	
2F -30	2F	47–48	47	0A	10	0A	10	**Pointer:** Start of Arrays
	30		48	10	16	08	8	
31 -32	31	49–50	49	0A	10	0A	10	**Pointer:** End of Arrays
	32		50	10	16	08	8	
33 -34	33	51–52	51	00	0	00	0	**Pointer:** String Storage (moving down)
	34		52	1E	30	A0	160	
35 -36	35	53–54	53	00	0	00	0	**Pointer:** String Utility
	36		54	1E	30	A0	160	
37 -38	37	55–56	55	00	0	00	0	**Pointer:** Limit of Memory
	38		56	1E	30	A0	160	
39 -3A	39	57–58	57	00	0	00	0	Current BASIC line number
	3A		58	FF	255	FF	255	
3B -3C	3B	59–60	59	00	0	00	0	Previous BASIC line number
	3C		60	00	0	00	0	
3D -3E	3D	61–62	61	3D	61	00	0	**Pointer:** BASIC statement for CONT
	3E		62	00	0	00	0	
3F -40	3F	63–64	63	00	0	00	0	Current DATA line number
	40		64	00	0	00	0	
41 -42	41	65–66	65	00	0	00	0	Current DATA address
	42		66	10	16	08	8	
43 -44	43	67–68	67	00	0	00	0	Input vector
	44		68	00	0	00	0	
45 -46	45	69–70	69	41	65	41	65	Current variable name
	46		70	00	0	00	0	
47 -48	47	71–72	71	05	5	05	5	Current variable address
	48		72	10	16	08	8	
49 -4A	49	73–74	73	05	5	05	5	Variable pointer for FOR/NEXT
	4A		74	10	16	08	8	
4B -4C	4B	75–76	75	00	0	00	0	Y–save; op–save; BASIC pointer save
	4C		76	00	0	00	0	
4D	4D	77	77	00	0	00	0	Comparison symbol accumulator
4E -53	4E	78–83	78	00	0	00	0	Misc. work area, pointers, etc.
	4F		79	00	0	00	0	
	50		80	00	0	00	0	
	51		81	00	0	00	0	
	52		82	00	0	00	0	
	53		83	03	3	03	3	
54 -56	54	84–86	84	4C	76	4C	76	Jump vector for functions
	55		85	0D	13	0D	13	
	56		86	D8	216	B8	184	
57 -60	57	87–96	87	00	0	00	0	Misc. numeric work area
	58		88	0A	10	0A	10	
	59		89	1F	15	07	7	
	5A		90	03	3	03	3	
	5B		91	1F	15	07	7	
	5C		92	00	0	00	0	
	5D		93	00	0	00	0	
	5E		94	00	0	00	0	
	5F		95	03	3	03	3	
	60		96	10	16	08	8	
61	61	97	97	87	135	87	135	Accum#1: Exponent
62 -65	62	98–101	98	00	0	00	0	Accum#1: Mantissa
	63		99	00	0	00	0	
	64		100	00	0	00	0	
	65		101	65	101	65	101	
66	66	102	102	4C	76	4C	76	Accum#1: Sign
67	67	103	103	00	0	00	0	Series evaluation constant pointer
68	68	104	104	00	0	00	0	Accum#1 hi–order (overflow)
69 -6E	69	105–110	105	00	0	00	0	Accum#2: Exponent
	6A		106	00	0	00	0	Accum#2: Mantissa
	6B		107	00	0	00	0	
	6C		108	00	0	00	0	
	6D		109	00	0	00	0	
	6E		110	00	0	00	0	Accum#2: Sign
6F	6F	111	111	00	0	00	0	Sign comparison, Acc#1 vs #2
70	70	112	112	00	0	00	0	Accum#1 lo–order (rounding)
71 -72	71	113–114	113	01	1	01	1	Cassette buff len/Series pointer
	72		114	01	1	01	1	
73 -8A	73	115–138	115	E6	230	E6	230	CHRGET subroutine; get BASIC char
	74		116	7A	122	7A	122	;INC $7A
	75		117	D0	208	D0	208	;BNE $0079
	76		118	02	2	02	2	
	77		119	E6	230	E6	230	;INC $7B
	78		120	7B	123	7B	123	
	79		121	AD	173	AD	173	;LDA $022D 64: LDA $022C
	7A		122	2D	45	2C	44	
	7B		123	02	2	02	2	
	7C		124	C9	201	C9	201	;CMP #$3A
	7D		125	3A	58	3A	58	
	7E		126	B0	176	B0	176	;BCS $008A
	7F		127	0A	10	0A	10	
	80		128	C9	201	C9	201	;CMP #$20
	81		129	20	32	20	32	
	82		130	F0	240	F0	240	;BEQ $0073
	83		131	EF	239	EF	239	
	84		132	38	56	38	56	;SEC
	85		133	E9	233	E9	233	;SBC #$30
	86		134	30	48	30	48	
	87		135	38	56	38	56	;SEC
	88		136	E9	233	E9	233	;SBC #$D0
	89		137	D0	208	D0	208	
	8A		138	60	96	60	96	;RTS
7A -7B	7A	122–123	122	2D	45	2C	44	BASIC pointer (within subrtn)
	7B		123	02	2	02	2	
8B -8F	8B	139–143	139	80	128	80	128	RND seed value
	8C		140	4F	79	4F	79	
	8D		141	C7	199	C7	199	
	8E		142	52	82	52	82	
	8F		143	58	88	58	88	
90	90	144	144	00	0	00	0	Status word ST
91	91	145	145	FF	255	FF	255	Keyswitch PIA: STOP and RVS flags
92	92	146	146	00	0	00	0	Timing constant for tape
93	93	147	147	00	0	00	0	LOAD = 0, VERIFY = 1
94	94	148	148	55	85	55	85	Serial output: deferred char flag
95	95	149	149	FF	255	FF	255	Serial deferred character
96	96	150	150	00	0	00	0	Tape EOT received
97	97	151	151	10	16	00	0	Register save
98	98	152	152	01	1	01	1	How many open files
99	99	153	153	00	0	00	0	Input device, normally 0
9A	9A	154	154	08	8	08	8	Output CMD device, normally 3
9B	9B	155	155	00	0	00	0	Tape character parity
9C	9C	156	156	00	0	00	0	Byte–received flag
9D	9D	157	157	80	128	80	128	Direct = $80/RUN = 0 output control
9E	9E	158	158	00	0	00	0	Tp Pass 1 error log/char buffer
9F	9F	159	159	00	0	00	0	Tp Pass 2 err log corrected
A0 -A2	A0	160–162	160	00	0	00	0	Jiffy Clock HML
	A1		161	25	37	3B	59	

Location				Contents				Description
Hex		Dec		VIC Hex	VIC Dec	C64 Hex	C64 Dec	
	A2		162	74	116	38	56	
A3	A3	163	163	55	85	55	85	Serial bit count/EOI flag
A4	A4	164	164	00	0	00	0	Cycle count
A5	A5	165	165	00	0	00	0	Countdown, tape write/bit count
A6	A6	166	166	00	0	00	0	Tape buffer pointers
A7	A7	167	167	00	0	00	0	Tp Wrt ldr count/Rd pass/inbit
A8	A8	168	168	00	0	00	0	Tp Wrt new byte/Rd error/inbit cnt
A9	A9	169	169	00	0	00	0	Wrt start bit/Rd bit err/stbit
AA	AA	170	170	00	0	00	0	Tp Scan;Cnt;Ld;End/byte assy
AB	AB	171	171	00	0	00	0	Wr lead length/Rd checksum/parity
AC-AD	AC	172-173	172	00	0	00	0	**Pointer:** tape bufr, scrolling
	AD		173	00	0	00	0	
AE-AF	AE	174-175	174	00	0	00	0	Tape end adds/End of program
	AF		175	00	0	00	0	
B0-B1	B0	176-177	176	00	0	00	0	Tape timing constants
	B1		177	00	0	00	0	
B2-B3	B2	178-179	178	3C	60	3C	60	**Pointer:** Start of Tape Buffer
	B3		179	03	3	03	3	
B4	B4	180	180	00	0	00	0	1=Tp timer enabled; bit count
B5	B5	181	181	00	0	00	0	Tp EOT/RS232 next bit to send
B6	B6	182	182	00	0	00	0	Read character error/outbyte buf
B7	B7	183	183	11	17	10	16	# characters in file name
B8	B8	184	184	05	5	05	5	Current logical file
B9	B9	185	185	65	101	65	101	Current secndy address
BA	BA	186	186	08	8	08	8	Current device
BB-BC	BB	187-188	187	EF	239	F0	240	Pointer to file name
	BC		188	1D	29	9F	159	
BD	BD	189	189	00	0	00	0	Wr shift word/Rd input char
BE	BE	190	190	00	0	00	0	# blocks remaining to Wr/Rd
BF	BF	191	191	00	0	00	0	Serial word buffer
C0	C0	192	192	00	0	00	0	Tape motor interlock
C1-C2	C1	193-194	193	00	0	00	0	I/O start address
	C2		194	20	32	A0	160	
C3-C4	C3	195-196	195	6D	109	30	48	Kernal setup pointer
	C4		196	FD	253	FD	253	
C5	C5	197	197	40	64	40	64	Last key pressed
C6	C6	198	198	00	0	00	0	# chars in keybd buffer
C7	C7	199	199	00	0	00	0	Screen reverse flag
C8	C8	200	200	4A	74	49	73	End-of-line for input pointer
C9-CA	C9	201-202	201	04	4	03	3	Input cursor log (row, column)
	CA		202	4A	74	49	73	
CB	CB	203	203	40	64	40	64	Which key: 64 if no key
CC	CC	204	204	01	1	01	1	0=flash cursor
CD	CD	205	205	0D	13	11	17	Cursor timing countdown
CE	CE	206	206	20	32	20	32	Character under cursor
CF	CF	207	207	00	0	00	0	Cursor in blink phase
D0	D0	208	208	00	0	00	0	Input from screen/from keyboard

Location				Contents				Description
Hex		Dec		VIC Hex	VIC Dec	C64 Hex	C64 Dec	
D1-D2	D1	209-210	209	C6	198	40	64	Pointer to screen line
	D2		210	1E	30	05	5	
D3	D3	211	211	00	0	00	0	Position of cursor on above line
D4	D4	212	212	00	0	00	0	0=direct cursor, else programmed
D5	D5	213	213	15	21	27	39	Current screen line length
D6	D6	214	214	09	9	08	8	Row where cursor lives
D7	D7	215	215	0D	13	0D	13	Last inkey/checksum/buffer
D8	D8	216	216	00	0	00	0	# of INSERTS outstanding
D9-F0	D9	217-240	217	9E	158	84	132	Screen line link table
	DA		218	9E	158	84	132	
	DB		219	9E	158	84	132	
	DC		220	9E	158	84	132	
	DD		221	9E	158	84	132	
	DE		222	9E	158	84	132	
	DF		223	1E	30	84	132	
	E0		224	1E	30	05	5	
	E1		225	1E	30	85	133	
	E2		226	9E	158	85	133	
	E3		227	9E	158	85	133	
	E4		228	9E	158	85	133	
	E5		229	9F	159	85	133	
	E6		230	9F	159	86	134	
	E7		231	9F	159	86	134	
	E8		232	9F	159	86	134	
	E9		233	9F	159	86	134	
	EA		234	9F	159	86	134	
	EB		235	9F	159	86	134	
	EC		236	9F	159	86	134	
	ED		237	9F	159	87	135	
	EE		238	9F	159	87	135	
	EF		239	9F	159	87	135	
	F0		240	9F	159	87	135	
F1	F1	241	241	FF	255	87	135	Dummy screen link
F2	F2	242	242	08	8	87	135	Screen row marker
F3-F4	F3	243-244	243	6E	110	F0	240	Screen colour pointer
	F4		244	96	150	D8	216	
F5-F6	F5	245-246	245	5E	94	81	129	Keyboard pointer
	F6		246	EC	236	EB	235	
F7-F8	F7	247-248	247	00	0	00	0	RS-232 Rcv pntr
	F8		248	00	0	00	0	
F9-FA	F9	249-250	249	00	0	00	0	RS-232 Tx pntr
	FA		250	00	0	00	0	
FB	FB	251	251	00	0	00	0	Not Known
FC	FC	252	252	00	0	00	0	Not Known
FD	FD	253	253	00	0	00	0	Not Known
FE	FE	254	254	00	0	00	0	Not Known
FF	FF	255	255	00	0	20	32	Start of Floating to ASCII Work Area

00FF-010A	256-266	Floating to ASCII work area
0100-013E	256-318	Tape error log
0100-01FF	256-511	Processor stack area
0200-0258	512-600	BASIC input buffer
0259-0262	601-610	Logical file table
0263-026C	611-620	Device number table
026D-0276	621-630	Sec address table
0277-0280	631-640	Keybd buffer
0281-0282	641-642	Start of BASIC Memory
0283-0284	643-644	Top of BASIC Memory
0285	645	Serial bus timeout flag
0286	646	Current colour code
0287	647	Colour under cursor
0288	648	Screen memory page
0289	649	Max size of keybd buffer
028A	650	Repeat all keys
028B	651	Repeat speed counter
028C	652	Repeat delay counter
028D	653	Keyboard Shift/Control flag
028E	654	Last shift pattern
028F-0290	655-656	Keyboard table setup pntr
0291	657	Keyboard shift mode
0292	658	0=scroll enable
0293	659	RS-232 control reg
0294	660	RS-232 command reg

		* Commodore 64 only	
0295-0296	661-662	Bit timing	
0297	663	RS-232 status	
0298	664	# bits to send	
0299-029A	665-666	RS-232 speed/code	
029B	667	RS232 receive pointer	
029C	668	RS232 input pointer	
029D	669	RS232 transmit pointer	
029E	670	RS232 output pointer	
029F-02A0	671-672	IRQ save during tape I/O	
02A1	673	CIA 2 (NMI) Interrupt control	*
02A2	674	CIA 1 Timer A control log	*
02A3	675	CIA 1 Interrupt log	*
02A4	676	CIA 1 Timer A enabled flag	*
02A5	677	Screen row marker	*
02C0-02FE	704-766	(Sprite 11)	*
0300-0301	768-769	Error message link	
0302-0303	770-771	BASIC warm start link	
0304-0305	772-773	Crunch BASIC tokens link	
0306-0307	774-775	Print tokens link	
0308-0309	776-777	Start new BASIC code link	
030A-030B	778-779	Get arithmetic element link	
030C	780	SYS A-reg save	*
030D	781	SYS X-reg save	*
030E	782	SYS Y-reg save	*

030F	783	SYS status reg save			*
0310-0312	784-786	USR function jump		64: (B248)	*
0314-0315	788-789	Hardware interrupt vector	20: (EABF)	64: (EA31)	
0316-0317	790-791	Break interrupt vector	20: (FED2)	64: (FE66)	
0318-0319	792-793	NMI interrupt vector	20: (FEAD)	64: (FE47)	
031A-031B	794-795	OPEN vector	20: (F40A)	64: (F34A)	
031C-031D	796-797	CLOSE vector	20: (F34A)	64: (F291)	
031E-031F	798-799	Set-input vector	20: (F2C7)	64: (F20E)	
0320-0321	800-801	Set-output vector	20: (F309)	64: (F250)	
0322-0323	802-803	Restore I/O vector	20: (F3F3)	64: (F333)	
0324-0325	804-805	INPUT vector	20: (F20E)	64: (F157)	
0326-0327	806-807	Output vector	20: (F27A)	64: (F1CA)	
0328-0329	808-809	Test-STOP vector	20: (F770)	64: (F6ED)	
032A-032B	810-811	GET vector	20: (F1F5)	64: (F13E)	
032C-032D	812-813	Abort I/O vector	20: (F3EF)	64: (F32F)	
032E-032F	814-815	Warm start vector		64: (FE66)	*
032E-032F	814-815	USR vector	20: (FED2)		
0330-0331	816-817	LOAD link	20: (F549)	64: (F4A5)	
0332-0333	818-819	SAVE link	20: (F685)	64: (F5ED)	
033C-03FB	828-1019	Cassette buffer			
0340-037E	832-894	(Sprite 13)			*
0380-03BE	896-958	(Sprite 14)			*
03C0-03FE	960-1022	(Sprite 15)			*

		VIC 20
0400-0FFF	1024-4095	3K RAM expansion area
1000-1FFF	4096-8191	Normal BASIC memory
1E00-1FF9	7680-8185	Normal Screen memory
1000-11F9	4096-4601	Screen memory w/expansion
1200-	4608-	BASIC memory w/expansion
2000-7FFF	8192-32767	Memory expansion area
8000-8FFF	32768-36863	Character bit maps
9000-900F	36864-36879	Video Interface Chip
9110-912F	37136-37151	VIA Interface - NMI
9120-912F	37152-37167	VIA Interface - IRQ
9400-95FF	37888-38399	Alternate Colour Nybble area
9600-97FF	38400-38911	Main Colour Nybble area
A000-BFFF	40960-49151	Plug-in ROM area
C000-FFFF	49152-65535	ROM: BASIC and Operating System
FF8A-FFF5	65418-65525	Jump Table

		Commodore 64
0400-07FF	1024-2047	Screen memory
0800-9FFF	2048-40959	BASIC RAM memory
8000-9FFF	32768-40959	Alternate: ROM plug-in area
A000-BFFF	40960-49151	ROM: BASIC
A000-BFFF	49060-49151	Alternate: RAM
C000-CFFF	49152-53247	RAM memory, including alternate
D000-D02E	53248-53294	Video Chip (6566)
D400-D41C	54272-54300	Sound Chip (6581 SID)
D800-DBFF	55296-56319	Color nybble memory
DC00-DC0F	56320-56335	Interface chip 1, IRQ (6526 CIA)
DD00-DD0F	56576-56591	Interface chip 2, NMI (6526 CIA)
D000-DFFF	53248-53294	Alternate: Character set
E000-FFFF	57344-65535	ROM: Operating System
E000-FFFF	57344-65535	Alternate: RAM
FF81-FFF5	65409-65525	Jump Table

B Series Memory Map

The following information applies to B systems released after April 1973, which contain a revised Machine Language Monitor.
(If SYS 6 doesn't bring in a monitor display complete with a 'period' prompt, it's the wrong version).

Notable features as compared to previous Commodore products include:

- CHRGOT is no longer in RAM. "Wedge" type coding must be inserted at links $029E and $02A0 .. which is likely to make the job easier.
- BASIC vectors have "split" - now, for example, there are discrete "Start of Variables" and "End of Variables", distinct from End of BASIC and Start of Arrays. Three-byte vectors (including bank number) are not uncommon.
- The "Jump Table" at top of memory is still accessible and reasonbly consistent with previous Commodore products.
- Simple machine language programs will fit into the spare 1k of ROM at $0400-0800 without trouble. Large programs must be implemented either by plug-in memory (RAM or ROM) in bank 15, or placed into another bank (preferably bank 3); supplementary code will be needed to make all the coding components fit.

The following map contains BASIC addresses specific to the B256/80; references to banks 0 to 4 are also specific to that machine. Most of the map is of general usage, however.

All Banks:		
0000	0	6509 Execution Register
0001	1	6509 Indirection Register
Bank 0: Unused.		
Bank 1:		
0002 - F000	2-61439	BASIC Program (text) RAM
FA5E - FB00	61440-64512	Input buffer area
Bank 2:		
0002 - FFFF	2-65535	BASIC Arrays in RAM.
Bank 3:		
0002 - 7FFF	2-32767	Unused RAM.
8000 - FFFF	32768-65535	BASIC Variables in RAM.
Bank 4:		
0002 - FBFF	2-64511	BASIC Strings (top down) in RAM
FC00 - FCFF	64512-64767	Unused RAM (descriptors?)
FD00 - FFFF	64768-65535	Current KEY definitions.
Banks 5 to 14. Unused.		
Bank 15:		
0002 - 0004	2-4	USR Jump
0005 - 0008	5-8	TI$ Output Elements: H,M,S,T
0009 - 000B	9-11	Pointer: Print Using Format
000C	12	Search Character
000D	13	Scan-between-Quotes Flag
000E	14	Input point; # subscripts
000F	15	Catalog line counter
0010	16	Default DIM flag
0011	17	Type: 255 = string, 0 = integer
0012	18	Type: 128 = integer, 0 = fl point
0013	19	Crunch flag
0014	20	Subscript index
0015	21	Input = 0; Get = 64; Read = 152
0016 - 0019	22-25	Disk status work values
001A	26	Current IO device fr prompt suppress
001B - 001C	27-28	Integer value
001D - 001F	29-31	Descriptor stack pointers
0020 - 002B	32-43	Misc work pointer
002D - 002E	45-46	**Pointer:** Start of BASIC
002F - 0030	47-48	**Pointer:** End of BASIC
0031 - 0032	49-50	**Pointer:** Start of Variables
0033 - 0034	51-52	**Pointer:** End of Variables
0035 - 0036	53-54	**Pointer:** Start of Arrays
0037 - 0038	55-56	**Pointer:** End of Arrays
0039 - 003A	57-58	**Pointer:** Variable work
003B - 003C	59-60	**Pointer:** Bottom of Strings
003D - 003E	61-62	**Pointer:** Utility String
003F - 0041	63-65	**Pointer:** Top of String Memory
0042 - 0043	66-67	Current BASIC line number
0044 - 0045	68-69	Old BASIC line number
0046 - 0047	70-71	Old BASIC text pointer
0049 - 004A	73-74	Data line number
004B - 004C	75-76	Data text pointer
004D - 004E	77-78	Input pointer
004F - 0050	79-80	Variable name
0051 - 0053	81-83	Variable address
0054 - 0056	84-86	For-loop pointer
0057 - 0058	87-88	Text pointer save
005A	90	Comparison symbol accumulator
005B - 005D	91-93	Function location
005E - 0060	94-96	Working string vector
0061 - 0063	97-99	Function jump code
0064 - 006E	100-110	Work pointers, values
006F	111	Exponent sign
0070	112	Accum string prefix
0071	113	Accum#1: Exponent
0072 - 0075	114-117	Accum#1: Mantissa
0076	118	Accum#1: Sign
0077	119	Series Evaluation Const pointer
0078	120	Accum#1 Hi order (overflow)
0079 - 007E	121-126	Accum#2, Ex, Man, Sign
007F	127	Sign comparison, Acc#1 vs #2
0080	128	Acc#1 Lo order (rounding)
0081 - 0084	129-132	Series, Work pointers
0085 - 0087	133-135	**Pointer:** BASIC text
0088 - 0089	136-137	Input pointer
008B - 008E	139-142	DOS parser work values
008F	143	Error type number
0090 - 0092	144-146	Pointer to file name
0093 - 0095	147-149	**Pointer:** Tape Buffer, Scrolling
0096 - 0098	150-152	Load end address/End of program
0099 - 009B	153-155	I/O start address
009C	156	Status word ST
009D	157	File name length
009E	158	Current logical file
009F	159	Current device
00A0	160	Current secondary address
00A1	161	Input device, normally 0
00A2	162	Output CMD device, normally 3
00A6 - 00A8	166-168	INBUF
00A9	169	Keyswitch PIA .. stop key,etc.
00AA	170	IEEE deferred flag
00AB	171	IEEE deferred character
00AC - 00AD	172-173	Segment transfer rtn vector
00AE - 00B3	174-179	Monitor register save
00B4	180	Monitor stack pointer save
00B5	181	Monitor bank number save
00B7 - 00B8	183-184	Monitor IRQ save/pointer
00B9 - 00BA	185-186	Monitor memory pointer
00BB - 00BC	187-188	Monitor secondary pointer
00BD	189	Monitor counter
00BE	190	Monitor misc byte
00BF	191	Monitor device number
00C0 - 00C1	192-193	Prog Key Table address
00C2 - 00C3	194-195	Programmable key address
00C4 - 00C7	196-199	Pointers to change Prog Key Table
00C8 - 00C9	200-201	Pointer to screen line
00CA	202	Screen line number
00CB	203	Positition of cursor on line
00CC	204	0 = text mode, else graphics md
00CD	205	Key pressed: 255 if no key
00CE	206	Old cursor column
00CF	207	Old cursor row
00D0	208	New character flag
00D1	209	# keys in Keyboard buffer
00D2	210	Quotes Flag
00D3	211	Insert key counter
00D4	212	Cursor type flag
00D5	213	Screen line length
00D6	214	# keys in 'key' buffer
00D7	215	Key repeat delay
00D8	216	Key repeat speed
00D9 - 00DA	217-218	Temporary Variables
00DB	219	Current output character
00DC	220	Top line of current screen
00DD	221	Bottom line of screen
00DE	222	Left edge of current screen
00DF	223	Right edge of screen
00E0	224	Keys: 255 = none;127 = key,111 = shift
00E1	225	Key: 255 = none (no shift)
00E2 - 00E5	226-229	Line Wrap Bits
0100	256	Hex to binary staging area
0100 - 010A	256-266	Numeric to ASCII work area
0100 - 01FE	256-510	Stack area
01FF	511	Stack pointer save location
0200 - 020F	512-527	File name area
0210 - 0226	528-550	Disk command work area
0255 - 0256	597-598	Misc work values for WAIT, etc
0257	599	'Bank' value
0258	600	Output logical file (CMD)
0259	601	Sign of TAN
025A - 025D	602-605	Pickup subrtn; misc work values
025E - 0276	606-630	PRINT USING working variables
0280 - 0281	640-641	Error routine link (854D)
0282 - 0283	642-643	Warm start link (85C5)
0284 - 0285	644-645	Crunch token link (88A9)
0286 - 0287	646-647	List link (89DB)
0288 - 0289	648-649	Command dispatch link (874C)
028A - 028B	650-651	Token evaluate link (969C)
028C - 028D	652-653	Expression eval link (95AF)
028E - 028F	654-655	CHRGOT link (B98E)
0290 - 0291	656-657	CHRGET vector (B994)
0292 - 0293	658-659	Float-fixed vector (B980)
0294 - 0295	660-661	Fixed-Float vector (9CA5)
0296 - 0297	662-663	Error trap vector
0298 - 0299	664-665	Error line number
029A - 029B	666-667	Error exit pointer
029C	668	Stack pointer save
029D - 029F	669-671	Temporary TRAP, DISPOSE bytes
02A0 - 02A5	672-677	Temporary INSTR$ bytes
02A6 - 02A7	678-679	Bank offset
0300 - 0301	768-769	IRQ vector (FBE9)
0302 - 0303	770-771	BRK vector (EE21)
0304 - 0305	772-773	NMI vector (FCAA)
0306 - 0307	774-775	OPEN vector (F6BF)
0308 - 0309	776-777	CLOSE vector (F5ED)
030A - 030B	778-779	Connect-input vector (F549)
030C - 030D	780-781	Connect-output vector (F5A3)
030E - 030F	782-783	Restore deflt I/O vector (F6A6)
0310 - 0311	784-785	Input vector (F49C)
0312 - 0313	786-787	Output vector (F4EE)
0314 - 0315	788-789	Stop key test vector (F96B)
0316 - 0317	790-791	GET vector (F43D)
0318 - 0319	792-793	Abort all files vector (F67F)
031A - 031B	794-795	Load vector (F746)
031C - 031D	796-797	Save vector (F84C)
031E - 031F	798-799	Monitor command vector (EE77)
0320 - 0321	800-801	Keyboard control vector (E01F)
0322 - 0323	802-803	Print control vector (E01F)
0324 - 0325	804-805	IEEE send LSA vector (F274)
0326 - 0327	806-807	IEEE send TSA vector (F280)
0328 - 0329	808-809	IEEE receive byte vector (F30A)
032A - 032B	810-811	IEEE send char vector (F297)
032C - 032D	812-813	IEEE send untalk vector (F2AB)
032E - 032F	814-815	IEEE send unlisten vector (F2AF)
0330 - 0331	816-817	IEEE send listen vector (F234)
0332 - 0333	818-819	IEEE send talk vector (F230)
0334 - 033D	820-829	File logical addresses table
033E - 0347	830-839	File device table
0348 - 0351	840-849	File secondary adds table
0352 - 0354	850-852	Bottom of system memory
0355 - 0357	853-855	Top of system memory
0358 - 035A	856-858	Bottom of user memory
035B - 035D	859-861	Top of user memory
035E	862	IEEE timeout; 0 = enabled
035F	863	0 = Load; 128 = Verify
0360	864	Number of open files
0361	865	Message mode byte
0363 - 0366	867-870	Misc register save bytes
0369	873	Timer toggle
036A - 036B	874-875	Cassette vector (dead end)
036F - 0371	879-881	Relocation start address
0375	885	Cassette motor flag (unused)
0376 - 0377	886-887	RS-232 Control, Command
037A	890	RS-232 Status
037B	891	RS-232 Handshake input
037C	892	RS-232 Input pointer
037D	893	RS-232 Arrival pointer
0380 - 0381	896-897	**Pointer:** Top of Memory
0382	898	Bank byte
0383	899	RVS flag
0384	900	Current line length
0385	901	Temp output char save
0386	902	0 = normal, 255 = auto insert
0387	903	0 = scrolling, 255 = no scroll
0388	904	Misc work byte for screen
0389	905	Index to prog key
038A	906	Scroll mode flag
038B	907	Bell mode flag
038C	908	Indirect bank save
038D - 03A0	909-928	Lengths of 'key' words
03A1 - 03AA	929-938	Bit mapped Tab stops
03AB - 03B4	939-948	Keyboard input buffer
03B5 - 03B6	949-950	'Key' word link (E91B)
03F8 - 03F9	1016-1017	Restart vector
03FA - 03FB	1018-1019	Restart test mask
0400 - 07FF	1024-2047	Free RAM (reserved for DOS)
0800 - 0FFF	2048-4095	Reserved for plug in RAM
1000 - 1FFF	4096-8191	Reserved for plug in DOS ROM
2000 - 7FFF	8192-23767	Reserved for cartridges
8000 - BFFF	32768-49151	BASIC ROM
C000 - CFFF	49152-53247	Unused
D000 - D7CF	53248-55247	Screen RAM
D800 - D801	55296-55297	Video controller 6545
DA00 - DA1C	55808-55836	Sound Interface Device 6581
DB00 - DB0F	56064-56079	Complex Interface Adaptor 6526
DC00 - DC0F	56320-56335	Complex Interface Adaptor 6526
DD00 - DD03	56576-56579	Asynchronous Comms IA 6551
DE00 - DE07	56832-56839	Tri Port Interface Adaptor 6525
DF00 - DF07	57088-57095	Tri Port Interface Adaptor 6525
E000 - FFFF	57344-65535	Kernal ROM

6525 Tri Port

DE00	NRFD	NDAC	EOI	DAV	ATN	RFN			56832
DE01	Sense	Cassette Motor	Out	ARB	Network Rx	Tx	SRQ	IFC	56833
DE02									56834
DE03	Data Direction Register For DE00								56835
DE04	Data Direction Register For DE01								56836
DE05	IRQ:			ACIA	IP	ALM	IEEE	PWR	56837
DE06	CB		CA: Graphics				IRQ Stack On		56838
DE07	Active Interrupt Register								56839

6525 Tri Port 2

DF00	Keyboard		57088
DF01	Select		57089
DF02	CRT Mode	Keyboard Read	57090
DF03	Data Direction Register for DF00 (out)		57091
DF04	Data Direction Register for DF01 (out)		57092
DF05	Data Direction Register for DF02 (in)		57093
DF06	Unused		57094

Commodore B128 ROM Routines

The following is a map of routines and data within the current (September 1983) version of the Commodore B128 computer. Caution: The same routines exists in the B256 but the addresses are not exactly the same.

Address	Routine
8000	Jumps: Warm start, Cold start
8006	Mask: 'CBM8'
800B	Reference Vectors (unused)
8027	Action vectors
803B	Action (run etc) vectors
80A3	Function vectors
80D1	Operation vectors
80EF	Keywords
828F	Message vectors
82E7	Messages
8550	Print 'Out of memory'
8552	Error routine
85AE	Print line number
85C0	Warm start
85F3	Handle new line
86A4	Rechain lines
86A3	Receive input line
871F	Find BASIC line
8751	Command dispatcher
87DB	Peek stack for FOR/GOSUB
8815	Open text space
8866	Stack too deep?
8889	Check string space
8890	Check BASIC space
889F	Check array space
88AB	'out of array space'
88BF	Crunch tokens
898D	Perform [LIST]
8A29	Perform [NEW]
8A45.	Perform [CLR]
8A90	'USING' characters
8A94	Perform [FOR]
8B06	Perform [NEXT]
8B79	Perform [RESTORE]
8BA8	Perform [STOP]
8BAA	Perform [END]
8BE9	Perform [CONT]
8C07	Perform [RUN]
8C25	Perform [GOSUB]
8C42	Perform [IF]
8C77	Perform [REM/ELSE]
8C7C	Perform [GO]
8C84	Perform [GOTO]
8CB8	Perform [RETURN]
8CDF	Perform [DATA]
8CED	Next statement
8CF0	Next line
8D16	Perform [TRAP]
8D2B	Perform [ON]
8D4E	Get fixed point number
8D8A	Perform [LET]
8DC4	Perform [RESUME]
8E24	Perform [DISPOSE]
8E7A	Perform [PRINT#]
8E80	Perform [CMD]
8E9D	Perform [PRINT]
8F15	Perform [GET]
8F4B	Perform [INPUT#]
8F66	Perform [INPUT]
8FA8	Prompt & input
8FEA	Perform [READ]
90E7	Perform [SYS]
910C	Perform [DIM]
9116	Perform [DEF]
9146	Perform [POKE]
9152	Perform [WAIT]
917F	Perform [KEY]
91BC	Perform [VERIFY]
91C8	Perform [LOAD]
921B	Perform [SAVE]
9243	Perform [OPEN]
9297	Perform [CLOSE]
92A1	Perform [CATALOG]
936D	Perform [DOPEN]
937E	Perform [APPEND]
93A9	Perform [DCLOSE]
93C3	Perform [DSAVE]
93CE	Perform [DLOAD]
93DE	Perform [BANK]
93EC	Perform [BSAVE]
940E	Perform [BLOAD]
9427	Perform [HEADER]
9464	Perform [SCRATCH]
949E	Perform [RECORD]
950A	Perform [DCLEAR]
9513	Perform [COLLECT]
952A	Perform [COPY]
9546	Perform [CONCAT]
9552	Perform [RENAME]
9560	Perform [BACKUP]
9586	Patch area
95C1	Evaluate expression
95CF	Recursive entry
96CB	Value of PI in binary
96F8	Evaluate [NOT]
9724	Eval within parens
979A	Go for disk status
986B	Evaluate [OR]
986E	Evaluate [AND]
98A8	Evaluate [COMPARE]
992C	Get var name/loc
99BF	Check alphabetic
9AF5	Array pntr subrtn
9B06	Float-fixed
9BA4	'bad subscript'
9BA7	'illegal quantity'
9CF5	Evaluate [FRE]
9D33	Evaluate [POS]
9D39	Fixed-float
9D4A	Confirm not direct
9D57	Check direct mode
9E07	Evaluate [PEEK]
9E30	Evaluate [subtract]
9E4D	Evaluate [add]
9F5E	Overflow error
9FCA	Evaluate [LOG]
A00B	Evaluate [multiply]
A0D0	+ 10 floating
A0E9	Evaluate [divide]
A148	Error: 'division by zero'
A210	Evaluate [SGN]
A22F	Evaluate [ABS]
A2B1	Evaluate [INT]
A3B4	Print numeric
A3C3	Print canned message
A50D	+ 32768
A537	Evaluate [SQR]
A541	Evaluate [power]
A57A	Evaluate [negate]
A5B3	Evaluate [EXP]
A659	Evaluate [RND]
A6A6	Evaluate [COS]
A6AD	Evaluate [SIN]
A6F8	Evaluate [TAN]
A791	Evaluate [ATN]
A7C0	Perform [PUDEF]
A7DB	Evaluate [STR$]
A805	Set up string descriptors
A81F	Scan and set up string
A8AB	Build string into memory
A8E6	Discard unwanted string
A955	..clean descriptor stack
AAD1	Evaluate [CHR$]
AAEB	Evaluate [LEFT$]
AB22	Evaluate [RIGHT$]
AB42	Evaluate [MID$]
AB8E	Evaluate [LEN]
AB9D	Evaluate [ASC]
ABAE	Evaluate [VAL]
AD53	Allocate dynamic string space
ADB5	Garbage collection
AF9D	Perform [DELETE]
AFF4	Get line range
B026	Perform <PRINT USING>
B4B8	Reset text pointer
B4E5	Evaluate integer
B501	Evaluate numeric
B504	Check numeric mode
B506	Check string mode
B52E	Print formal character
B53A	Print character
B7CB	Disk command formats
BA1E	Float-fixed conversion
BA26	CHRGET - Get new BASIC character
BA29	CHRGOT - Get previous character
BA50	Numeric check
BA5A	Set text bank
BA69	Set bank from FAC
BA6E	Set bank from $60
BA73	Set bank from $24
BA78	Set bank 15 (system)
BA7D	Set bank 4
BA82	Set bank 2
BA87	Set bank 3
BA8C	Set bank 1 (text)
BB82	Startup message
BBA6	Link vectors ($0280)
BBE1	BASIC I/O with error traps.
BBE2	Perform BASIC Open
BBE8	Perform BASIC Get
BBEE	Perform BASIC Input
BBF4	Perform BASIC output
BBFA	Perform BASIC connect-input
BC00	Perform BASIC connect-output
BC06	Perform BASIC Load
BC0C	Perform BASIC Save
BC12	Error on above BASIC I/O
BC1A	Output error message
E000	**Kernal:**
E24D	Set graphics mode
E251	Set text mode
E260	Set up CRT controlr
E299	Output to screen
E306	Control key fanout
E311	Escape key vector
E314	Cursor up/down
E331	Cursor left/right
E344	Rvs/rvs off
E34A	Home/clear
E35A	Tab & tab set/clear
E394	Carriage return
E3B4	Move screen line
E48D	Ring bell
E4BA	Delete numeric
E4F5	Check start of line
E51E	Locate line wrap
E532	Goto start of line
E544	Goto end of line
E5AE	Delete/insert
E61C	Initiate load/run
E655	Escape key link
E658	Insert a line
E66D	Delete a line
E694	Erase right
E6A9	Erase left
E6BD	Scroll up
E6BF	Scroll down
E6E3	Enable scrolling
E6E5	Disable scrolling
E7BE	Create new prog key
E949	Get prog key addrs
E970	Escape sequence
E979	Cancel escape seq
E985	Escape key vectors
E9B9	Set top/left
E9BB	Set bottom/right
E9BC	Set window
E9C7	Set full screen
E9D6	Enable bell
E9D8	Disable bell
E9DC	Set underline mode
E9E6	Set flashing cursor
E9EC	Set solid cursor
E9EF	Set non-flashing cursor
E9F6	Reverse screen
E9F9	(alternate characters)
EA05	Un-reverse screen
EA08	(normal characters)
EA20	Cancel auto insert
EA23	Set auto insert
EBA9	Load/run keys
EBB3	Screen line adds low
EBCB	Screen line adds high
EBE4	Control key vectors
EC24	Default 'key' word lengths
EC2E	Default 'key' words
EC67	Bit masks
EC6F	CRT controller setup
EE00	Monitor trap
EE09	Monitor call (60937)
EE21	BRK entry
EE55	Monitor reentry
EED5	Monitor vectors
EEF9	Perform [.X] exit to BASIC
EEFF	Set PC address
EF08	Set register address
EF17	Print prompt group
EF1F	Print space
EF22	Print question mark
EF27	Monitor prompt
EF31	Register heading
EF4C	Perform [.R] register display
EF8F	Perform [.M] memory display
EFCB	Perform [.;] register change
EFE1	Perform [.V] bank switch
EFEB	Perform [.U]
EFF5	Perform [..] memory change
F010	Perform [.G] go
F04A	Perform [.L/.S] load/save
F0F6	Print 2 hex bytes
F0FB	Print hex byte
F107	Print hex digit
F113	Swap temp1/temp2
F123	Get 4 hex digits
F130	Get hex byte
F154	ASCII hex to binary
F15F	Input character
F165	Perform [.@] disk status
F1C3	Error messages
F221	Print error message
F230	Send 'talk'
F234	Send 'listen'
F236	Send IEEE command
F274	Send Listen SA
F277	Release ATN
F280	Send Talk SA
F283	Prepare IEEE in
F297	Send IEEE deferred
F2AB	Send 'untalk'
F2AF	Send 'unlisten'
F2B9	Send IEEE byte
F30A	Receive IEEE byte
F381	Open RS-232
F3C7	Convert to true ASCII
F3DC	Convert to PETSCII
F400	Allocate buffer
F4EE	Output
F549	Connect input
F5A3	Connect output
F5ED	Close file
F63E	Find file LA
F650	Set file details
F660	Find matching SA
F678	Search for file
F67F	Abort all files
F6A6	Restore default I/O
F6BF	Open file
F707	Open IEEE
F746	Load
F84C	Save
F8E6	Read time of day
F90E	Set TOD/alarm
F939	File error entry points
F997	Power up reset
FAFD	Vectors
FB31	NMI entry
FB34	Set function addrs
FB43	Set file parameters
FB4A	Read status byte
FB5A	Set message mode
FB5F	Log into status byte
FB74	Set timeout
FB78	Set/read top of memory
FB8D	Set/read bottom of memory
FBA2	Set page 3 vectors
FBD6	IRQ interrupt
FBE9	Interrupt routines
FC9F	Wind up interrupt
FE9D	Exsub - Bank Transfer Sequences:
FF04	.excomm
FF19	..ipinit
FF24	..putaxs
FF2A	..putas
FF6C	Jumbo jump table
FFF6	Bank transfer execution
FFFA	Hard vectors

6526 CIA 1

Address	Contents	Decimal
DB00	Inter-Processor Data	56064
DB01	X / IRQ Out / X / X / SEMAPH / Busy	56065
DB02	Data Direction Register For DB00	56066
DB02	Data Direction Register For DB01	56067
	Unused	
DB0D	IP Flag	56077
DB0E	Unused	56078
DB0F	Unused	56079

6526 CIA 2

Address	Contents	Decimal
DC00	IEEE Data In/Out	56320
DC01	User Port	56321
DC02	Data Direction Register For DC00	56322
DC02	Data Direction Register For DC01	56323
	Unused	
DC06	Timer B — L	56326
DC07	Timer B — H	56327
DC08	Time Of Day Clock (TOD) — 1/10 Sec.	56328
DC09	Time Of Day Clock (TOD) — Sec.	56329
DC0A	Time Of Day Clock (TOD) — Min.	56330
DC0B	Time Of Day Clock (TOD) — Hour	56331
DC0C	Unused	56332
DC0D	Alarm	56333
DC0E	Unused	56334
DC0F	TOD / Timer Force / Timer Start	56335

6551 ACIA

Address	Contents	Decimal
DD00	Data Register	
DD01	IRQ / DSR / DCD / Ready (Tx, Rx) / OV / Error (FR, PA)	56577
DD02	XTRR Stop / # of Bits / Clk / Speed	56578
DD03	Parity / Echo / IRQ (Tx, Rx) / DTR	56579

6545 CRT Controller

D800 55296	D801 55297	Typical Value (Decimal)
0	Horizontal Total	108 or 126 or 127
1	Horizontal Char Displayed	80
2	Horizontal Sync Position	83 or 98 or 96
3	Sync Width V H	15 or 10
4	Vertical Total	25 or 31 or 38
5	Vert Total Adjust	3 or 6 or 1
6	Vertical Displayed	25
7	Vert. Sync Position	25 or 28 or 30
8	Mode	0
9	Scan Lines	13 or 7
10	Cursor Start	96 (blink) or 0 or 6 (underline)
11	Cursor End	13 or 7
12	Display Address H	0
13	Display Address L	0
14	Cursor Address H	Varies
15	Cursor Address L	Varies
16	Light Pen In H	0
17	Light Pen In L	0

Most Register are Write Only   14/15 are Read/Write
16/17 are Read Only
Registers 10, 14 and 15 change as the cursor moves

6581 SID

Address	Contents	Decimal
DA01	Voice 1 Frequency High	55809
DA04	Saw Tooth / Ring Mod / Key	55812
DA05	Attack / Decay	55813
DA06	Sustain / Release	55814
DA0F	Voice 3 Modulating Freq Hi	55823
DA18	Volume	55832

Commodore 16 / Plus 4 RAM Memory Map

(Preliminary: September 25/84. Note that the previously available locations for VIC/C64, $00FC to $00FF, are no longer available.

Hex	Decimal	Description						
0000	0	Chip directional register	00B6–00B7	182–183	**Pointer:** start of tape buffer	04C6	1222	Subroutine (bank via $6F)
0001	1	Chip I/O; serial bus/cassette	00B8–00B9	184–185	Misc. pointer	04D1	1233	Subroutine (bank via $5F)
0002	2	Loop type match	00BA–00BB	186–187	Cassette I/O work pointer	04DC	1244	Subroutine (bank via $64)
0003–0006	3–6	Renumber parameters	00BC–00C1	188–193	Work pointers	04E7–04EA	1255–1258	PU characters (,.$)
0007	7	Search character	00C2	194	Screen reverse flag	04EB–04EE	1259–1262	String work area
0008	8	Scan-quotes flag	00C3	195	End-of-line for input pointer	04EF–04F6	1263–1270	TRAP and error flags
0009	9	TAB column save	00C4–00C5	196–197	Input cursor log (row, column)	04F7	1271	Stack pointer for error trap
000A	10	0=LOAD, 1=VERIFY	00C6	198	Which key: 64 if no key	04F8–04FB	1272–1275	DO loop work area
000B	11	Input buffer pointer / # of subscripts	00C7	199	Input from screen/from keyboard	04FC–04FF	1276–1279	Sound work area
000C	12	Default DIM flag	00C8–00C9	200–201	Pointer to screen line	0500–0502	1280–1282	USR program jump
000D	13	Type: FF=string; 00=numeric	00CA	202	Position of cursor on above line	0503–0508	1283–1288	RND seed value
000E	14	Type: 80=integer; 00=floating point	00CB	203	0=direct cursor; else programmed	0509–0512	1289–1298	Logical file table
000F	15	DATA scan/LIST quote/memory flag	00CC	204	Current screen line length	0513–051C	1299–1308	Device number table
0010	16	Subscript/FNx flag	00CD	205	Row where cursor lives	051D–0526	1309–1318	Secondary address table
0011	17	0=INPUT;$40=GET;$98=READ	00CE	206	Last I/O character	0527–0530	1319–1328	Keyboard buffer
0012	18	ATN sign/Comparison evaluation flag	00CF	207	Number of INSERTs outstanding	0531–0532	1329–1330	Start of BASIC memory
0013	19	Current I/O prompt flag	00D0–00D7	208–215	Unused; reserved for speech	0533–0534	1331–1332	Top of BASIC memory
0014–0015	20–21	Integer value	00D8–00E8	216–232	Unused	0535–0536	1333–1334	Timeout/end flags, not used much
0016	22	**Pointer:** temporary string stack	00E9	233	Work value	0537–0538	1335–1336	Tape buffer counts, not used much
0017–0018	23–24	Last temporary string vector	00EA–00EB	234–235	Color line pointer	0539	1337	Tape buffer pointer
0019–0021	25–33	Stack for temporary strings	00EC–00EE	236–238	Screen work values	053A	1338	Tape file type
0022–0025	34–37	Utility pointer area	00EF	239	Number of characters in keyboard buffer	053B	1339	Character (color) attribute
0026–002A	38–42	Product area for multiplication	00F0	240	Screen freeze flag	053C	1340	Flash flag
002B–002C	43–44	**Pointer:** Start-of-BASIC	00F1–F4	241–244	Monitor work values	053D	1341	Unused
002D–002E	45–46	**Pointer:** Start-of-variables	00F5	245	Cassette checksum	053E	1342	Screen page (unused)
002F–0030	47–48	**Pointer:** Start-of-arrays	00F6	246	Monitor work value	053F	1343	Keyboard buffer size
0031–0032	49–50	**Pointer:** End-of-arrays	00F7–00F8	247–248	Cassette work values	0540	1344	Key repeat: 128=all, 64=none
0033–0034	51–52	**Pointer:** String-storage (moving down)	00F9	249	DMA control mask	0541–0542	1345–1346	Key repeat counters
0035–0036	53–54	Utility string pointer	00FA	250	Work byte	0543	1347	Key shift flag
0037–0038	55–56	**Pointer:** Limit-of-Memory	00FB	251	Current ROM bank	0544	1348	Key font interlock flag
0039–003A	57–58	Current BASIC line number	0100–01FF	256–511	Processor stack area	0545–0546	1349–1350	Key input vector (DB7A)
003B–003C	59–60	Textpointer: BASIC work point	0200–0258	512–600	BASIC input buffer	0547	1351	Text/Graphics mode lockout flag
003D–003E	61–62	**Pointer:** BASIC stack for CONT	0259–025A	601–602	Previous Basic line number	0548	1352	Scroll enable flag
003F–0040	63–64	Current DATA line number	025B–025C	603–604	**Pointer:** Basic statement for CONT	0549–054A	1353–1354	Screen work values
0041–0042	65–66	Current DATA address	025D–02AC	605–684	DOS command work area	054B–0551	1355–1372	MLM work locations
0043–0044	67–68	Input vector	02AD–02B0	685–688	Graphics cursor, X and Y	0552–0557	1362–1367	MLM registers (PC/SR/A/X/Y)
0045–0046	69–70	Current variable name	02B1–02B4	689–692	Graphics working cursor	0558–055C	1368–1372	MLM work locations
0047–0048	71–72	Current variable address	02B5–02CB	693–715	Graphics work area	055D	1373	FN key pending count
0049–004A	73–74	Variable pointer for FOR/NEXT	02CC–02E8	716–744	Print-using, graphics work area	055E	1374	FN key pointer
004B–004C	75–76	Y-save; op-save; BASIC pointer save	02E9	745	Temp screen row number	055F–05E6	1375–1510	Key definition area
004D	77	Comparison symbol accumulator	02EA	746	String length	05E7–05EB	1511–1515	DMA work locations
004E–0053	78–83	Misc. work area, pointers, and so on	02EB	747	255=Trace on	05EC–05EF	1516–1519	ROM ID (PAT) table
0054–0056	84–86	Jump vector for functions	02EC–02EE	748–750	Directory work area	05F0–05F1	1520	Long Jump vector
0057–0060	87–96	Miscellaneous numeric work area	02EF	751	Graphics work area	05F2–05F4	1522–1524	Long Jump registers
0061	97	Accum#1: exponent	02F0	752	Number of graphics parameters	05F5–06EB	1524–1791	Reserved RAM for extra ROMs
0062–0065	98–101	Accum#1: mantissa	02F1	753	Parameter relative (1) or absolute (0)	06EC–07AF	1792–1967	BASIC pseudo-stack
0066	102	Accum#1: sign	02F2–02F3	754–755	Float-fixed vector	07B0–07CC	1968–1996	Tape working values
0067	103	Series evaluation constant pointer	02F4–02F5	756–757	Fixed-float vector	07CD–07D0	1997–2000	RS232 working values
0068	104	Accum#1 hi-order (overflow)	02F6–02FD	758–765	Unused	07D1	2001	RS232 in pointer
0069–006E	105–110	Accum#2: exponent, and so on	02FE–02FF	766–767	Reserved for cartridge vector	07D2	2002	RS232 read pointer
006F	111	Sign comparison, Acc#1 versus #2	0300–0301	768–769	Error message link [8686]	07D3	2003	RS232 input counter
0070	112	Accum#1 lo-order (rounding)	0302–0303	770–771	BASIC warm start link [8712]	07D4–07D8	2004–2008	RS232 work values
0071–0072	113–114	Cassette buffer len/Series pointer	0304–0305	772–773	Crunch BASIC tokens link [8956]	07D9–07E4	2009–2020	Character load program
0073–0074	115–116	Auto line number increment	0306–0307	774–775	Print tokens link [8B6E]	07E5	2021	Current screen bottom margin
0075	117	Graphics flag	0308–0309	776–777	Start new BASIC code link [8BD6]	07E6	2022	Current screen top margin
0076–0079	118–123	Misc work values	030A–030B	778–779	Get arithmetic element link [9417]	07E7	2023	Current screen left margin
007C–007D	124–125	BASIC pseudo-stack pointer	030C–030D	780–781	Crunch hook vector [896A]	07E8	2024	Current screen right margin
007E–008F	126–143	Misc work values	030E–030F	782–783	List hook vector [8B88]	07E9	2025	0=Scrolling enabled
0090	144	Status word ST	0310–0311	784–785	Execute hook vector [8C8B]	07EA	2026	255=Auto Insert enabled
0091	145	Keyswitch IA: STOP and RVS flags	0312–0313	786–787	Interrupt link (CE42)	07EB	2027	Previous character printed
0094	148	Serial output: deferred character flag	0314–0315	788–789	IRQ vector (CE0E)	07EC–07ED	2028–2029	Current (color) attribute
0095	149	Serial deferred character	0316–0317	790–791	Break interrupt vector (F44C)	07EE–07F1	2030–2033	Screen line wrap table
0096	150	Register save	0318–0319	792–793	OPEN vector (EF53)	07F2	2034	SYS A-reg save
0097	151	How many open files	031A–031B	794–795	CLOSE vector (EE5D)	07F3	2035	SYS X-reg save
0098	152	Input device, normally 0	031C–031D	796–797	Set-input vector (ED18)	07F4	2036	SYS Y-reg save
0099	153	Output CMD device, normally 3	031E–031F	798–799	Set-output vector (ED60)	07F5	2037	SYS status reg save
009A	154	Direct=$80/RUN=0 output control	0320–0321	800–801	Restore I/O vector (EF0C)	07F6	2038	New key detect
009B–009C	155–156	**Pointer:** tape buffer, scrolling	0322–0323	802–803	Input vector (EBE8)	07F7	2039	Lockout Ctrl-S
009D–009E	157–158	End of program pointer	0324–0325	804–805	Output vector (EC4B)	07F8	2040	Monitor read: ROM or RAM
009F–00A0	159–160	Work area	0326–0327	806–807	Test-STOP vector (F265)	07F9	2041	Color decode switch
00A1–00A2	160–161	Monitor working vector	0328–0329	808–809	GET vector (EBD9)	07FA	2042	Split screen bit mask
00A3–00A5	163–165	Jiffy Clock HML	032A–032B	810–811	Abort I/O vector (EF08)	07FB	2043	Split screen video base
00A6	166	Serial bit count/EOI flag	032C–032D	812–813	USR vector (F44C)	07FC	2044	Tape motor interlock
00A7	167	Tape shift byte	032E–032F	814–815	LOAD vector (F04A)	0800–0BE7	2048–3047	Color memory
00A8	168	Serial cycle count	0330–0331	816–817	SAVE vector (F1A4)	0C00–0FE7	3072–4071	Screen memory
00A9	169	Temporary color vector	0332–03F2	818–1010	Cassette buffer	1000–FFFF	4096–65535	BASIC RAM memory (normal)
00AA	170	Countdown,tape write/bit count	03F3–03F6	1011–1014	Tape write/read counters	2000–FFFF	8192–65535	BASIC RAM memory (hi-res)
00AB	171	Number of characters in file name	03F7–0436	1015–1078	RS232 input buffer	8000–FFFF	32768–65535	ROM: BASIC
00AC	172	Current logical file	0437–0472	1079–1138	Tape error log	D000–D7FF	53248–55295	Character sets in ROM
00AD	173	Current secondary address	0473	1139	CHRGET subroutine	FD00–FD0F	64768–64783	ACIA communications chip
00AE	174	Current device	0479	1145	CHRGOT subroutine	FD10–FD1F	64784–64799	Parallel port/6529
00AF–00B0	175–176	Pointer to file name	0494	1172	Subroutine (self banking)	FDD0–FDDF	64976–64991	ROM bank select (write only)
00B1	177	Tape error count	04A5	1189	Subroutine (bank via $3B)	FE00–FEFF	65024–65279	DMA disk interface
00B2–00B3	178–179	I/O start address	04B0	1200	Subroutine (bank via $22)	FF00–FF1F	65280–65311	TED I/O control chip
00B3–00B4	180–181	Load address pointer	04BB	1211	Subroutine (bank via $24)	FF3E–FF3F	65342–65343	ROM/RAM select (write only)

Commodore 16 / Plus 4 ROM Memory Map

Address	Description
8000	C-16 ROM start
8003	Warm start
8019	Basic setup
802A	Fix/float vectors
802E	Intialize Basic
80BC	CHRGET pointers
80C2	Print Basic start msg
8105	Page 3 vectors
8123	CHRGET copy
818E	Keywords
8383	Action vectors
8415	Function vectors
8453	Dfunct vectors
8471	Messages
866F	Print 'READY.'
8683	Error routine
870F	Ready for Basic
872E	Handle new line
8818	Rechain lines
885A	Receive input line
8871	Scan Basic-stack
8905	Expand Basic-stack
8953	Crunch tokens
8A3D	Find Basic line
8A79	Perform [NEW]
8A93	Run
8A98	Perform [CLR]
8AED	PUDEF characters
8AF1	Back up text pointer
8AFF	Perform [LIST]
8BBC	Perform [RUN]
8C9A	Perform [RESTORE]
8CD8	Perform [STOP]
8CDA	Perform [END]
8D03	Perform [CONT]
8D2C	Perform [GOSUB]
8D4D	Perform [GOTO]
8D83	Perform [RETURN]
8DB0	Perform [DATA]
8DBE	Scan for next statement
8DC1	Scan for next line
8DE1	Perform [IF]
8E0B	Perform [REM/ELSE]
8E1B	Perform [ON]
8E3E	Get fixed point number
8E7C	Perform [LET]
8FE0	Perform [PRINT#]
8FE6	Perform [CMD]
9000	Perform [PRINT]
9088	Print from (y a)
90A5	Print format char
90B8	Perform [GET]
90EE	Perform [INPUT#]
9108	Perform [INPUT]
9142	Prompt and input
914F	Perform [READ]
9294	Perform [NEXT]
9314	Check type match
932C	Evaluate expression
9471	Fixed-float conversion
9485	Eval within parens
94AD	Search for variable
95F8	Evaluate <OR>
95FB	Evaluate <AND>
9628	Evaluate <COMPARE>
969B	Perform [DIM]
96A5	Locate variable
973A	Check alphabetic
9744	Create variable
985B	Array pointer subroutine
9871	Float-fixed conversion
989B	Set up array
9A2F	Compute array size
9A62	Evaluate <FRE>
9A76	Fixed-float
9A7D	Evaluate <POS>
9A86	Check direct
9A9D	Perform [DEF]
9ACB	Check FN syntax
9ADE	Perform [FN]
9B54	Set up string descriptor
9B66	Evaluate <STR$>
9B70	Calculate string vector
9B74	Set up string
9BDA	Concatenate
9C1B	Build string into memory
9C4B	Discard unwanted string
9C52	Make room for string
9CAA	Clean descriptor stack
9CBB	Evaluate <CHR$>
9CCF	Evaluate <LEFT$>
9D03	Evaluate <RIGHT$>
9D15	Evaluate <MID$>
9D46	Pull string params
9D61	Evaluate <LEN>
9D67	Exit string mode
9D70	Evaluate <ASC>
9D81	Input byte parameter
9D93	Evaluate <VAL>
9DD2	Get params for POKE/WAIT
9DDE	Get params for SOUND
9DE4	Convert to fixed point
9DFA	Evaluate <PEEK>
9E12	Perform [POKE]
9E1B	Evaluate <DEC>
9E6A	Perform [WAIT]
9E87	Evaluate <subtract>
9E9E	Evaluate <add>
9F7B	Complement FAC#1
9FB7	Multiply by zero byte
A01E	Evaluate <LOG>
A07B	Evaluate <multiply>
A0A9	Multiply a bit
A0DC	Memory to FAC#2
A107	Memory to FAC#2
A137	Adjust FAC#1/#2
A154	Under/overflow
A162	Multiply by ten
A183	Divide by ten
A197	Evaluate <divide>
A21F	Memory to FAC#1
A24C	FAC#1 to memory
A281	FAC#2 to FAC#1
A291	FAC#1 to FAC#2
A2A0	Round FAC#1
A2B0	Get sign
A2BE	Evaluate <SGN>
A2CE	Fixed-float
A2DD	Evaluate <ABS>
A2E0	Compare FAC#1 to memory
A327	Float-fixed
A358	Evaluate <INT>
A37F	String to FAC#1
A453	Print 'IN..'
A45A	Print number
A46F	Float to ASCII
A5E4	Evaluate <SQR>
A5EE	Evaluate <power>
A627	Evaluate <negative>
A660	Evaluate <EXP>
A6B3	Series evaluation 1
A6C9	Series evaluation 2
A707	Evaluate <RND>
A760	Save Basic-stack
A769	Restore Basic-stack
A772	Trim Basic-stack
A77D	Kernal calls
A7B5	Perform [SYS]
A7CF	SYS return
A7DE	Perform [SAVE]
A7F0	Perform [VERIFY]
A7F3	Perform [LOAD]
A84D	Perform [OPEN]
A85A	Perform [CLOSE]
A86B	Params for LOAD/SAVE
A89D	Check default parameters
A8A5	Check comma
A8A8	Params for OPEN/CLOSE
A906	Allocate string space
A954	Garbage collection
AA57	Calculate end of string
AA70	Evaluate <COS>
AA77	Evaluate <SIN>
AAC0	Evaluate <TAN>
AB1A	Evaluate <ATN>
AB8F	Perform [RENUMBER]
ADCA	Perform [FOR]
AE5A	Perform [DELETE]
AEF7	Print using
B42B	Perform [TRAP]
B440	Perform [RESUME]
B4BE	Evaluate <ERR$>
B507	Evaluate <HEX$>
B544	Perform [PUDEF]
B557	Perform [DO]
B5AC	Perform [EXIT]
B603	Perform [LOOP]
B652	Perform [TRON]
B655	Perform [TROFF]
B6CD	Perform [AUTO]
B6E8	Perform [HELP]
B729	Perform [KEY]
B849	Perform [SOUND]
B8BD	Perform [VOL]
B8D1	Perform [PAINT]
B9D4	Perform [CHAR]
BAE2	Perform [BOX]
BD35	Perform [GSHAPE]
BE29	Perform [SSHAPE]
BF79	Evaluate <RGR>
BF85	Evaluate <RCLR>
BF87	Evaluate <RLUM>
BFC1	Evaluate <JOY>
BFFD	Evaluate <RDOT>
C01E	Perform [CIRCLE]
C37B	Set graphics cursor
C3F7	Parse graphics command
C48F	Get graphics parameter
C4D9	Perform [DRAW]
C50F	Perform [LOCATE]
C51A	Perform [COLOR]
C567	Perform [SCNCLR]
C5B8	Perform [SCALE]
C5C3	Perform [GRAPHIC]
C7BF	Confirm graphics
C8BC	Perform [DIRECTORY]
C941	Perform [DSAVE]
C951	Perform [DLOAD]
C968	Perform [HEADER]
C99C	Perform [SCRATCH]
C9CC	Perform [COLLECT]
C9DA	Perform [COPY]
C9F4	Perform [RENAME]
CA00	Perform [BACKUP]
CB1F	Parse DOS command
CE00	Interrupt entry
CE0E	IRQ sequence
CE60	Do screen split
CEF0	Kernal - UDTIM
CF26	Kernal - RDTIM
CF2D	Kernal - SETTIM
CF8A	Get color mode
CF96	Fetch memory
CFBF	Handle tape motor
D000	Graphic character set
D400	Text character set
D802	Screen addresses
D834	Kernal - SCREEN
D839	Kernal - PLOT
D888	ESC-n normal screen
D8A1	Setup screen line
D9BA	Quote test
D9C7	Screen output wrap
D9D9	Setup screen print
DB11	Kernal - SCNKEY
DC41	Function keys
DC49	Output to screen
DC9B	ESC-O; key escape
DE06	Decode escapes
DE1A	ESC vectors
DE48	ESC-R; reduce screen
DE5E	ESC-T; top window
DE60	ESC-B; bottom window
DE8B	ESC-I; insert line
DEA0	ESC-D; delete line
DECB	ESC-Q; erase to end
DEE1	ESC-P; erase fm start
DEF6	ESC-V; scroll up
DF04	ESC-W; scroll down
DF1D	ESC-L; scroll enable
DF20	ESC-M; scroll disable
DF26	ESC-C; cancel insert
DF29	ESC-A; auto insert
DF39	Check screen line wrap
DF46	Break screen wrap
DF59	Make screen wrap
DF66	Calculate screen wrap mask
DF82	ESC-J; start-line
DF95	ESC-K; end-line
E01E	Keyboard sets
E153	Send 'Talk'
E156	Send 'Listen'
E181	Send to serial bus
E1E9	Serial timeouts
E1F7	Send listen SA
E1FC	Slear ATN
E203	Send talk SA
E20C	Wait for clock
E21D	Send serial deferred
E22F	Send 'Untalk'
E2B8	Serial clock on
E2BF	Serial clock off
E2C6	Serial output '1'
E2CD	Serial output '0'
E2D4	Get serial & clock
E2DC	Delay 1 ms
E319	Print 'Press play & rec'
E31B	Print 'Press play'
E38D	Start tape
E3B0	Kill motor
E3B7	Clear tape buffer
E3C3	Setup tape buffer
E413	Send tape cycle
E447	Send tape 'long'
E452	Send tape 'short'
E45D	Send tape 'medium'
E468	Send tape '0' bit
E474	Send tape '1' bit
E48C	Send tape byte
E535	Initiate tape write
E56C	Write tape header
E68E	Bit masks
E9CC	Find any tape header
EA21	Find specific header
EA5B	RS-232 out (IRQ)
EA95	RS-232 in (IRQ)
EBD9	Kernal - GETIN
EBE8	Kernal - CHRIN
EC0E	Get from tape
EC14	Get from RS-232
EC1C	Get from serial
EC4B	Kernal - CHROUT
EC63	Send to tape
EC84	Send to RS-232
EC8B	Kernal - ACPTR
ECDF	Kernal - CIOUT
ED18	Kernal - CHKIN
ED60	Kernal - CHKOUT
EDFA	Kernal - TALK
EE1A	Kernal - TKSA
EE2C	Kernal - LISTEN
EE4D	Kernal - SECOND
EE5D	Kernal - CLOSE
EF08	Kernal - CLALL
EF0C	Kernal - CLRCHN
EF23	Kernal - UNLSN
EF3B	Kernal - UNTLK
EF53	Kernal - OPEN
F005	Send SA
F043	Kernal - LOAD
F064	Load from serial
F0F0	Load from tape
F172	Print filename
F194	Kernal - SAVE
F1A4	• Save link •
F1B5	Save to serial
F228	Print 'SAVING'
F234	Save to tape
F265	Kernal - STOP
F2A4	System reset
F2CE	Transfer page 3 vectors
F2EB	Vectors page 3
F352	Identify 16K/32K/64K RAM
F3D2	Key lengths
F3DA	Key definitions
F40C	Kernal - SETNAM
F413	Kernal - SETLFS
F41A	Kernal - SETMSG
F41C	Kernal - READST
F41E	Change ST
F423	Kernal - SETTMO
F427	Kernal - MEMTOP
F42F	Set MEMTOP
F436	Kernal - MEMBOT -
F445	Perform [MONITOR]
F44C	BRK/USR entry
F478	Perform [.R]
F4D7	Perform [.M]
F50A	Perform [change reg]
F529	Perform [.>]
F54B	Perform [.G]
F570	Monitor commands
F580	Monitor vectors
F5CE	Perform [.C]
F5D1	Perform [.T]
F60E	Perform [.H]
F66E	Perform [.S/.L/.V]
F70A	Perform [.F]
F724	Perform [.D]
F83D	Op code mode
F881	Machine language codes
F89B	Mnemonics
F91F	Perform [.A]
FB72	Decrement $F1/2
FB86	Decrement $9F/A0
FB94	Increment $A1/2
FBB7	Save registers
FBC1	Recall registers
FC19	Kernal - IOBASE -
FC59	'Phoenix' routine
FC7F	Long Fetch routine
FC89	Long Jump routine
FCB3	IRQ entry
FCB8	Long IRQ routine
FCF1	'SRT' kernal entry
FCF4	'Phoenix' entry
FCF7	Long Fetch entry
FCFA	Long Jump entry
FCFD	Long IRQ entry
FF90	Jump table
FFFC	System vectors

+4 Kernal Jump Table

Label	Hex	Dec	Jumps to	Comments
CINT	FF81	65409	$D84E	initialize screen editor
IOINIT	FF84	65412	$F30B	initialize input/output
RAMTAS	FF87	65415	$F352	init ram/tapbuf/set screen
RESTOR	FF8A	65418	$F2CE	restore default i/o devices
VECTOR	FF8D	65421	$F2D3	store/restore ram vectors (c=0/1)
SETMSG	FF90	65424	$F41A	enable/disable 'kernal' messages
SECOND	FF93	65427	$EE4D	send sec address after listen
TKSA	FF96	65430	$EE1A	send sec addres after talk
MEMTOP	FF99	65433	$F427	read/set top of mem (c=1/0)
MEMBOT	FF9C	65436	$F436	read/set bottom of mem (c=1/0)
SCNKEY	FF9F	65439	$DB11	scan keyboard
SETTMO	FFA2	65442	$F423	set/reset ieee timeout (a<>127)
ACPTR	FFA5	65445	$EC8B	input byte from serial port
CIOUT	FFA8	65448	$ECDF	output byte to serial port
UNTLK	FFAB	65451	$EF3B	command serial bus to 'untalk'
UNLSN	FFAE	65454	$EF23	command serial bus to 'unlisten'
LISTEN	FFB1	65457	$EE2C	cmd devices on ser bus to 'listen'
TALK	FFB4	65460	$EDFA	cmd serial bus device to 'talk'
READST	FFB7	65463	$F41C	read i/o status word
SETLSF	FFBA	65466	$F413	set log/unit/sec addresses
SETNAM	FFBD	65469	$F40C	set file name
OPEN	FFC0	65472	($0318)	open a logical file
CLOSE	FFC3	65475	($031A)	close a specified logical file
CHKIN	FFC6	65478	($031C)	open channel for input
CHKOUT	FFC9	65481	($031E)	open channel for output
CLRCHN	FFCC	65484	($0320)	restore default i/o devices
CHRIN	FFCF	65487	($0322)	input character from channel
CHROUT	FFD2	65490	($0324)	output character to channel
LOAD	FFD5	65493	$F043	load/verify ram from device
SAVE	FFD8	65496	$F194	'save' ram to a device
SETTIM	FFDB	65499	$CF2D	set real time clock
RDTIM	FFDE	65502	$CF26	read real time clock
STOP	FFE1	65505	($0326)	scan stop key depressed
GETIN	FFE4	65508	($0328)	get char from current input dev
CLALL	FFE7	65511	($032A)	close all channels and files
UDTIM	FFEA	65514	$CEF0	increment real time clock
SCREEN	FFED	65517	$D834	return scr size in rows/columns
PLOT	FFF0	65520	$D839	read/ser cursor position (c=1/0)
IOBASE	FFF3	65523	$FC19	returns base add of i/o devices
	$FFFA	.BYT $A4/$F2		system nmi $F2A4
	$FFFC	.BYT $F6/$FF		system reset $FFF6
	$FFFE	.BYT $B3/$FC		system irq $FCB3

Ted Chip Register Map

Reg#	Address	7	6	5	4	3	2	1	0
0	FF00	Timer#1 Reload Value Bits 0-7 (Low)							
1	FF01	Timer#1 Reload Value Bits 8-15 (High)							
2	FF02	Timer#2 Reload Value Bits 0-7 (Low)							
3	FF03	Timer#2 Reload Value Bits 8-15 (High)							
4	FF04	Timer#3 Reload Value Bits 0-7 (Low)							
5	FF05	Timer#3 Reload Value Bits 8-15 (High)							
6	FF06	Test	ECM	BMM	Blank	# Rows	Y Offset		
7	FF07	Rvs Off	PAL	Freeze	MCM	# Cols	X Offset		
8	FF08	Keyboard Latch (IN and OUT)							
9	FF09	IRQ	T3	NC	T2	T1	LP	RAS	NC
10	FF0A	NC	IE-T3	NC	IE-T2	IE-T1	IE-LP	IE-RAS	RC 8
11	FF0B	Raster Compare (RC) Bits 7-0							
12	FF0C	NC	NC	NC	NC	NC	NC	CP 9	CP 8
13	FF0D	Cursor Position (CP) Bits 7-0							
14	FF0E	Sound 1 (S_1) Bits 7-0							
15	FF0F	Sound 2 (S_2) Bits 7-0							
16	FF10	NC	NC	NC	NC	NC	NC	S_2 9	S_2 8
17	FF11	Sound Reload	Noise	V2 Sel	V1 Sel	Volume			
18	FF12	NC	NC	Bit Map Base			ROM Bank	S_1 9	S_1 8
19	FF13	Character Base (5-0)						Single Clock	Status
20	FF14	Video Matrix (4-0)					NC	NC	NC
21	FF15	NC	Luminance 0			Colour 0			
22	FF16	NC	Luminance 1			Colour 1			
23	FF17	NC	Luminance 2			Colour 2			
24	FF18	NC	Luminance 3			Colour 3			
25	FF19	NC	Luminance 4			Colour 4			
26	FF1A	NC	NC	NC	NC	NC	NC	BRP 9	BRP 8
27	FF1B	Bit Map Raster Position (BRP) Bits 7-0							
28	FF1C	NC	NC	NC	NC	NC	NC	NC	VRP 8
29	FF1D	Vertical Raster Position (VRP) Bits 7-0							
30	FF1E	Horizontal Position (HP) Bits 7-0							
31	FF1F	NC	Blink Rate (3-0)				VSUB (2-0)		
62	FF3E	Write to select ROM access							
63	FF3F	Write to select RAM access							

NC = No Connection. IE = Interrupt Enable. Tn = Timer n.
BMM = Bit Map Mode. ECM = Ext Char Mode. MCM = Multi-Colour Mode

Disk Data File Format

Program Files	
Byte#	**Description**
0–1	Track/Sector pointer to next Program block
2–255	Up to 254 bytes of BASIC Program text. End-of-File is marked by three consecutive bytes of $00

Sequential and Relative Record Data	
Byte#	**Description**
0–1	Track/Sector pointer to next sequential data block
2–255	Up to 254 bytes of data

Notes: Track link of $00 in byte zero indicates last data block (Track 0 is not used by DOS). Sector link is then next byte position to receive data. End of relative record data indicated by ST = 64. Unused Record bytes are padded with CHR$(0). Relative File terminated with $FF.

Relative File Side Sector Format	
Byte#	**Description**
0–1	Track/Sector pointer to next Side Sector
2	8050/4040/2031/1541: Side Sector number 5250/D9060/D9090: constant $FE
3	Relative Record Length
4–5	Track/Sector pointer - First Side Sector
6–7	Track/Sector pointer - Second Side Sector
8–9	Track/Sector pointer - Third Side Sector
10–11	Track/Sector pointer - Fourth Side Sector
12–13	Track/Sector pointer - Fifth Side Sector
14–15	Track/Sector pointer - Sixth Side Sector
16–255	Track/Sector pointers to 120 data blocks. Total of 720 blocks (maximum 182.8 K Bytes) per file

DOS 2.7 and DOS 3.0 Super Side Sector contain Track/Sector pointers to 127 groups of 6 Side Sectors as above for maximum file size of 23.25 MB.

Disk Utility-Command Set

Command	Abbreviations	Format
Block-Read	B-R	"B-R:"lf;dr;t;s
Block-Write	B-W	"B-W:"lf;dr;t;s
Block-Execute	B-E	"B-E:"lf;dr;t;s
Buffer-Pointer	B-P	"B-P:"lf;p
Block-Allocate	B-A	"B-A:"dr;t;s
Block-Free	B-F	"B-F:"dr;t;s
Memory-Write	M-W	"M-W"adl/adh/nc/data
Memory-Read	M-R	"M-R"adl/adh/nc
Memory-Execute	M-E	"M-E"adl/adh
User	U	"Ux:"lf;dr;t;s

LF	The Logical File Number in the associated OPEN Statement
DR	The Drive Number: 0 (or 1 on dual drives)
T	The Track Number: 1 through 154 (depending on the model number)
S	The Sector Number: 0 through 192 (depending on the model number)
P	The pointer Position for the Buffer Pointer
ADL	The Low Byte of the Address (use CHR$(ADL))
ADH	The High Byte of the Address (use CHR$(ADH))
NC	The Number of Characters: 1 through 34
DATA	The actual data in hexadecimal. this is transmitted by using the CHR$ function, ie. CHR$(17) would send the decimal equivalent of hex 11
X	The index to the user table

Disk LED Error Diagnostics

Number of Flashes	4040		8050	
	Error Cause	Component, Location	Error Cause	Component, Location
1	Zero Page	6532, C1, E1	Zero Page	6532, C1, E1
2	ROM	H1	ROM	2364, L1
3	ROM	L1	ROM	2364, H1
4	ROM	J1	N/A	
5	Zero Page	6530, K3; 6504, H3	Zero Page	6530, K3; 6502, H3
6	N/A		N/A	
7	RAM	2114, D4, D5	RAM	2114, D4, D5
8	RAM	2114, E4, E5	RAM	2114, E4, E5
9	RAM	2114, F4, F5	RAM	2114, F4, F5
10	ROM	6530, K3; 6504, H3	ROM	6530, K3; 6502, H3

PET/CBM Disk Access Routines

Action	Hex	Dec	Method To Access From Within Basic
CONCAT	$FF93	65427	sys65427 "filename",d# to "otherfilename",d#
DOPEN	$FF96	65430	sys65430 #lf,"filename",d#
DCLOSE	$FF99	65433	sys65433 alone or followed by #lf
RECORD	$FF9C	65436	sys65436 #lf,(r#),(pr)
HEADER	$FF9F	65439	sys65439 "disk name",d#,iid
COLLECT	$FFA2	65442	sys65442 d#
BACKUP	$FFA5	65445	sys65445 d# to d#
COPY	$FFA8	65448	sys65448 "filename",d# to "filename",d#
APPEND	$FFAB	65451	sys65451 #lf,"filename"
DSAVE	$FFAE	65454	sys65454 "filename",d#
DLOAD	$FFB1	65457	sys65457 "filename",d#
CATALOG	$FFB4	65460	sys65460 d# (same for DIRECTORY)
RENAME	$FFB7	65463	sys65463 "filename",d# to "newfilename"
SCRATCH	$FFBA	65466	sys65466 "filename",d#
OPEN	$FFC0	65472	sys(65472) lf,ua,sa,"d#:filename,type,operation"
CLOSE	$FFC3	65475	sys(65475) lf
LOAD	$FFD5	65493	sys(65493) "d#:filename",ua
SAVE	$FFD8	65496	sys(65496) "d#:filename",ua
VERIFY	$FFDB	65499	sys(65499) "d#:filename",ua

lf = logical file number	pr = pointer within record
sa = secondary address	id = 2 character identifier
ua = drive unit address	type = either : s (seq), p (prg), or u (usr)
d# = drive number	operation = either : w (write), r (read),
r# = record number	a (append), or (m) modify

User Command Jump Table

Standard Syntax	Alternate (1541: n/a)	Function	
U0		Reset User Jump Vector	
U1	UA	Block-Read replacement	
U2	UB	Block-Write replacement	
		4040/8X50 2031/D90XX	**1541/2031 Low-Profile**
U3	UC	Jump to $1300	Jump to $0500
U4	UD	Jump to $1303	Jump to $0503
U5	UE	Jump to $1306	Jump to $0506
U6	UF	Jump to $1309	Jump to $0509
U7	UG	Jump to $130C	Jump to $050C
U8	UH	Jump to $130F	Jump to $050F
U9	UI	Jump to $10F0	Jump to $FFFA (NMI)
U:	UJ	Power-Up Vector (reset)	

Sector Distribution By Track

	Number of Sectors		
Track Number	**4040**	**2031**	**1541**
1 - 17	21	21	21
18 - 24	19	19	19
25 - 30	18	18	18
31 - 35	17	17	17

Track Number	8050	8250	
1 - 39	29	29	
40 - 53	27	27	
54 - 64	25	25	
65 - 77	23	23	
78 - 116		29	
117 - 130		27	
131 - 141		25	
142 - 154		23	

D9060/D9090 – 153 tracks per recording surface (4 on D9060 and 6 on the D9090) with 32 sectors per track

GCR Codes

GCR is the method in which disk data is magnetically stored. It is based on transitions (ie. 1 to 0, or 0 to 1) A transition is decoded as 0, no transition decodes to a 1.

Hex	GCR	Binary	Dec	Hex	GCR	Binary	Dec
$00	01010	0000	0	$08	01001	1000	8
$01	01011	0001	1	$09	11001	1001	9
$02	10010	0010	2	$0A	11010	1010	10
$03	10011	0011	3	$0B	11011	1011	11
$04	01110	0100	4	$0C	01101	1100	12
$05	01111	0101	5	$0D	11101	1101	13
$06	10110	0110	6	$0E	11110	1110	14
$07	10111	0111	7	$0F	[illegible]	[illegible]	[illegible]

4040 Disk Memory Map

4040 System Constants

Hex Val	Label	Description
$00	NOTRDY	i/o not ready
$00	RDMODE	open read mode
$01	ATNA	atn active
$01	LISNER	ieee listener flag
$01	RDYLST	i/o ready to listen
$01	SEQTYP	sequential file type
$01	VAL	job code for validate
$01	WTMODE	open write mode
$02	APMODE	open append mode
$02	DACO	data accepted – output
$02	DOSVER	dos version
$02	PRGTYP	program file type
$03	MDMODE	open modify mode
$03	USRTYP	usr file type
$04	NMODES	number of modes within table MODLST ('rwam')
$04	RELTYP	relative file type
$04	RFDO	ready for data – output
$05	MXFILS	maximum number of filenames in string
$05	NTYPES	number of file types from TYPLST ('dspur')
$06	CMDCHN	command channel = mxchns – 2
$06	NBCMDS	start for offset for comparison with table BCTAB ('afrwep')
$06	NSSL	number of side sector links
$07	CTBSIZ	command table size
$07	DIRTYP	direct file type
$07	ERRCHN	error channel number = mxchns – 1
$07	VERERR	controller verify error
$08	EOIO	eoi – output
$08	EOISND	not (eoi) to send
$08	LED1	active led 1
$08	MXCHNS	maximum number of channels
$08	PCMD	commands not parsed error
$0B	LDCMD	load command # / load command image
$0B	NCMDS	number of commands from CMDTBL ('ivdmbupcrsn')
$0C	BFCNT	available buffer count
$0C	MSGLEN	length of 'block free' message at FREMSG
$0D	CR	carriage return
$0E	TYPMSK	type mask for matching pattern type
$0F	CMDSA	command channel secondary address
$10	DAVO	data valid – output
$10	ERRSA	error channel secondary address
$10	LED0	active led 0
$10	SSIOFF	offset into side sector for data block pointers
$11	IRSA	internal read secondary address channel
$12	IWSA	internal write secondary address channel
$12	MAXSA	maximum secondary address
$18	DIRLEN	length of directory buffer
$1B	NBSIZ	NAMBUF text size
$1C	CBPTR	command buffer pointer
$1E	CMDIND	command index. 2
$20	EOII	eoi – input
$20	ERRLED	hardware initialization error led
$20	OVRFLO	overflow flag value
$24	MAXTRK	maximum track number
$30	BADSYN	error : general syntax
$31	BADCMD	error : invalid command
$32	LONGLN	error : long line
$33	BADFN	error : invalid filename
$34	NOFILE	error : no file given
$3A	CMDLEN	length of command buffer
$3F	LXINT	LINDX 0 to 5 free
$3F	UNLSN	IEEE unlisten command number
$40	DAVI	data valid – input
$40	NDACI	no data accepted – input
$41	FM2040	dos format version # for 2040 drive
$42	FM2030	dos format version # for 2030 drive
$50	NOREC	error : record not present
$51	RECOVF	error : overflow in record
$52	BIGFIL	error : file too large
$60	FILOPN	error : file open
$61	FILNOP	error : file not open
$62	FLNTFD	error : file not found
$63	FLEXST	error : file exists
$64	MISTYP	error : file type mismatch
$65	NOBLK	error : no block
$66	BADTS	error : illegal track or sector
$70	NOCHNL	error : no channel available
$71	DIRERR	error : directory error
$72	DSKFUL	error : diskette full
$73	CBMV2	'cbm dos v2.1 4040' message number
$78	NSSP	number of pointers in side sector
$80	ATNI	atn inactive
$80	EOIOUT	talk with eoi
$80	LRF	last record flag
$80	NRFDI	next record flag for drive 1
$80	READ	controller job type : read
$80	TALKER	ieee talker flag
$81	RNDEOI	random with eoi
$88	RDYTLK	talk no eoi
$89	RNDRDY	random chnrdy = rdytlk + rdylst
$90	WRITE	controller job type : write
$A0	WVERFY	controller job type : write/verify
$B0	SEEK	controller job type : seek
$C0	BUMP	controller job type : bump
$C4	ERRTOK	size of error message token table
$D0	JUMPC	controller job type : jump
$E0	EXEC	controller job type : execute

4040 RAM Memory Map with Zero Page Contents at Power Up

Hex Location		Content	CBM Label	Function
00-01	00	EA	USRJMP	User Jump Table Pointer ($FFEA)
	01	FF		
02-03	02	00	BMPNT	Bit Map Pointer
	03	00		
04-09	04	04	TEMP : T0	Temp Work Space - CMD Jump Table
	05	00	: T1	
	06	00	: T2	
	07	09	: T3	
	08	00	: T4	
	09	00		
0A-0B	0A	00	IP	Indirect Pointer Variable
	0B	40		
0C	0C	28	LSNADR	Listen Address : Device # + $20
0D	0D	48	TLKADR	Talker Address : Device # + $40
0E	0E	00	LSNACT	Active Listener Flag
0F	0F	00	TLKACT	Active Talker Flag
10	10	00	ADRSED	Addressed Flag
11	11	00	PRGTRK	Last Program Accessed
12	12	00	DRVNUM	Current Drive Number
13	13	00	TRACK	Current Track
14	14	00	SECTOR	Current Sector
15	15	06	LINDX	Logical Index
16	16	0F	SA	Current Secondary Address
17	17	6F	ORGSA	Original Secondary Address
18	18	3F	DATA	Temporary Data Byte
19	19	00	R0	Temp Work Area
1A	1A	00	R1	Temp Work Area
1B	1B	00	R2	Temp Work Area
1C	1C	00	R3	Temp Work Area
1D	1D	00	R4	Temp Work Area
1E-21	1E	00	RESULT	Result of Multiply/Divide Rtns.
	1F	00		
	20	00		
	21	00		
22-26	22	00	ACCUM	Remainder of Multiply/Divide Rtns.
	23	00		
	24	00		
	25	00		
	26	00		
27-28	27	05	DIRBUF	Pointer To Directory Buffer - $4305
	28	43		
29-48	29	00	BUFTAB	Buffer Byte Ptrs. 16 entries, 2 bytes each. point to current byte in corresponding buf.
	29	00		Buffer Byte Ptrs. : Buffer #0 Low
	2A	11		High
	2B	00		: Buffer #1 Low
	2C	12		High
	2D	00		: Buffer #2 Low
	2E	13		High
	2F	00		: Buffer #3 Low
	30	20		High
	31	00		: Buffer #4 Low
	32	21		High
	33	00		: Buffer #5 Low
	34	22		High
	35	00		: Buffer #6 Low
	36	23		High
	37	00		: Buffer #7 Low
	38	30		High
	39	00		: Buffer #8 Low
	3A	31		High
	3B	00		: Buffer #9 Low
	3C	32		High
	3D	00		: Buffer #10 Low
	3E	33		High
	3F	00		: Buffer #11 Low
	40	40		High
	41	00		: BAM Drive 0 Low
	42	41		: BAM Drive 0 High
	43	00		: BAM Drive 1 Low
	44	42		: BAM Drive 1 High
	45	00		: CMD Buffer Low
	46	43		: CMD Buffer High
	47	DD		: Error Output Buffer Low
	48	43		: Error Output Buffer High
49-50	49	FF	BUF0	Inactive Flags For Buffers. next 16 bytes
	4A	FF		store buffer pairs for double buffering
	4B	FF		blocks of seq files. bit7 = 1 indicates
	4C	FF		inactive buffer. direct access channels use
	4D	FF		only one buffer. 2nd entry is set to $FF
	4E	FF		indicating no buffer
	4F	0E		
	50	0F		
51-58	51	FF	BUF1	Active Flags For Buffers. second buffer
	52	FF		number pair associated with channel
	53	FF		
	54	FF		
	55	FF		
	56	FF		
	57	FF		
	58	FF		
59	59	00	NBKL	Number of Blocks Low
59-60	59	00	RECL	Low Record # To Find Relative File
	5A	00		
	5B	00		
	5C	00		
	5D	00		
	5E	00		
	5F	00		
	60	00		
61	61	00	NBKH	Number of Blocks - High Byte
61-68	61	00	RFCH	High Record # To Find Relative File
	62	00		
	63	00		
	64	00		
	65	00		
	66	00		
	67	00		
	68	00		
69-70	69	00	NR	Next Record Table
	6A	00		
	6B	00		
	6C	00		
	6D	00		
	6E	00		
	6F	00		
	70	00		
71-78	71	00	RS	Relative Record Size Table
	72	00		
	73	00		
	74	00		
	75	00		
	76	00		
	77	00		
	78	00		
79-80	79	FF		Side Sector Table
	7A	FF		
	7B	FF		
	7C	FF		
	7D	FF		
	7E	FF		
	7F	FF		
	80	FF		
81-	81	00	F1PTR	File Stream 1 Pointer
82	82	00	RECPTR	1st Byte Wanted From Relative File
83	83	00	SSNUM	Side Sector # Of Relative Record
84	84	00	SSIND	Index Into Side Sector
85	85	00	RELPTR	Ptr To 1st Byte Wanted In REL File
86-8A	86	00	FILENT	Directory Entry Of Located Files
	87	00		(Index-2) into sector
	88	00		Sector of track 18
	89	00		Bit Pattern : IIISSSSS
	8A	00		
8B-8F	8B	00	FILDAT	File Data
	8C	00		file type times 2 plus drive number
	8D	00		bit7 = 1 indicates search both drives
	8E	00		
	8F	00		
90-97	90	00	FILTYP	Channel File Type. 8 entries, 1 byte each.
	91	00		contains file type times 2 plus drive num.
	92	00		bit7 = 1 indicates search both drives
	93	00		SEQ = type 1
	94	00		PRG = type 2
	95	00		USR = type 3
	96	00		REL = type 4
	97	00		direct = type 7
98-9F	98	00	CHNRDY	Channel Status. 8 entries, 1 byte each.
	99	00		indicates channels status for IEEE talk and
	9A	00		listen sequences. bit7 = 1 channel is talker
	9B	00		to IEEE. bit3 = 0 send EOI on next byte
	9C	00		(talker only). bit0 = 1 channel is listener
	9D	00		to IEEE. other bits unused
	9E	01		
	9F	88		
A0	A0	20	EOIFLG	Temporary EOI
A1	A1	00	JOBNUM	Current Job Number
A2-B4	A2	FF	LINTAB	Logical Index Table contains corresponding
	A3	FF		secondary address associated with channel
	A4	FF		number. $FF indicates no active channel.
	A5	FF		bits 7 and 6 indicate channel direction:
	A6	FF		0 0 = read channel
	A7	FF		1 0 = write channel
	A8	FF		0 1 = read/write channel
	A9	FF		1 1 = no channel
	AA	FF		
	AB	FF		
	AC	FF		
	AD	FF		
	AE	FF		
	AF	FF		
	B0	FF		
	B1	86		
	B2	07		
	B3	FF		
	B4	FF		
B5-BC	B5	00	CHNDAT	Channel Data Byte. contians data byte for
	B6	00		output to IEEE through GET routines
	B7	00		
	B8	00		
	B9	00		
	BA	00		
	BB	00		
	BC	30		
DB-C4	BD	00	LSTCHR	Channel Last Character Pointer. last
	BE	00		character pointer is active buffer
	BF	00		associated with channel. = 0 if not last
	C0	00		block in SEQ file
	C1	00		
	C2	00		
	C3	00		
	C4	E7		
C5	C5	00	TYPE	Active File Type

** The Balance Of Zero Page Is Not Used Directly By DOS **

C6 = 00 C7 = 00
C8 = 00 C9 = 00 CA = 00 CB = 00 CC = 00 CD = 00 CE = 00 CF = 00

D0 = 00 D1 = 00 D2 = 00 D3 = 00 D4 = 00 D5 = 00 D6 = 00 D7 = 00
D8 = 00 D9 = 00 DA = 00 DB = 00 DC = 00 DD = 00 DE = 00 DF = 00

E0 = 00 E1 = 00 E2 = 00 E3 = 00 E4 = 00 E5 = 00 E6 = 00 E7 = 00
E8 = 00 E9 = 00 EA = 00 EB = 00 EC = 00 ED = 00 EE = 00 EF = 00

F0 = 00 F1 = 00 F2 = 00 F3 = 00 F4 = 00 F5 = 00 F6 = C8 F7 = B9
F8 = C8 F9 = D9 FA = 0D FB = DA FC = 6D FD = DB FE = B7 FF = D4

4040 RAM Memory $0100-

Location	Label	Description
0100-01FF		the stack
0200	IEEEDI	ieee data in
0201	PADD1	ieee data in direction
0202	IEEEDO	ieee data out
0203	PBDD1	ieee data out direction
0204		
0205		
0206		
0207		
0208-027F		unconnected
0280	PAD2	IEEE control port: **
0281	PADD2	**
0282	PBD2	**
0283	PBDD2	**
0284	ATNND	** atn is irq causing ???
0285	ATNPD	**
0286	ATNNE	**
0287	ATNPE	**
0288-0FFF		unconnected
1000	ID	Interrupt Delay (** start of shared memory **)
1001		motor acceleration delay
1002		motor cutoff time
1003-1011	JOBS que	buf #0 Job Codes are:
1004		buf #1 $80 - Read - read t & s specified
1005		buf #2 by header into data buf
1006		buf #3 $90 - Write - write t & s specified
1007		buf #4 by header from data buf
1008		buf #5 $A0 - Verify - compare t & s specified
1009		buf #6 by header with data buf
100A		buf #7 $B0 - Seek - find any header on track
100B		buf #8 specified by hdr. put in data buf
100C		buf #9 $C0 - Bump - track must be set to 1.
100D		buf #10 positions head to track 1
100E		buf #11 $D0 - Jump - jump to user ml code
100F		buf #12 in data buf
1010		buf #13 $E0 - Execute - same as Jump with
1011		buf #14 head in position and drive at speed
1012-1020	TRKS	jobs' track number. used by controller for quick reference to track #. must match track in corresponding header
1021-10xx	HDRS	job headers for buffers 0-14. 15 entries of 8 bytes each. controller calculates checksum upon execution of job. bits 6 and 7 are used as ID extension. currently set at 0 and 0
1021-1022	job header	buf #0 ID1, ID2 Job Error Codes
1023-1024		buf #0 track, sector returned into Job Que
1025-1026		buf #0 checksum, off after Job is executed
1027-1028		buf #0 spare1, spare2 No error : $01
1029-102A	job header	buf #1 ID1, ID2 Can't find header block : $02
102B-102C		buf #1 track, sector No sync character : $03
102D-102E		buf #1 checksum, off Data block not present : $04
102F-1030		buf #1 spare1, spare2 Chksum err in data blk : $05
1031-1032	job header	buf #2 ID1, ID2 not used : $06
1033-1034		buf #2 track, sector Verify error : $07
1035-1036		buf #2 checksum, off Write protect on : $08
1037-1038		buf #2 spare1, spare2 Chksum err in hdr : $09
1039-103A	job header	buf #3 ID1, ID2 Data ran into next hdr : $0A
103B-103C		buf #3 track, sector Disk id mismatch : $0B
103D-103E		buf #3 checksum, off Decoding error : $10
103F-1040		buf #3 spare1, spare2
1041-1048	job header	buf #4 ID1, ID2, trk, sec, chksum, off, 2 spares
1049-1050	job header	buf #5 ID1, ID2, trk, sec, chksum, off, 2 spares
1051-1058	job header	buf #6 ID1, ID2, trk, sec, chksum, off, 2 spares
1059-1060	job header	buf #7 ID1, ID2, trk, sec, chksum, off, 2 spares
1061-1068	job header	buf #8 ID1, ID2, trk, sec, chksum, off, 2 spares
1069-1070	job header	buf #9 ID1, ID2, trk, sec, chksum, off, 2 spares
1071-1078	job header	buf #10 ID1, ID2, trk, sec, chksum, off, 2 spares
1079-1080	job header	buf #11 ID1, ID2, trk, sec, chksum, off, 2 spares
1081-1088	job header	buf #12 ID1, ID2, trk, sec, chksum, off, 2 spares
1089-1090	job header	buf #13 ID1, ID2, trk, sec, chksum, off, 2 spares
1091-1098	job header	buf #14 ID1, ID2, trk, sec, chksum, off, 2 spares
1099-109E	NUMSEC	sectors/track table
109F	VERNUM	dos version number
10A0	ACTJOB	controller's active job
10A1-10EF		not used
10F0-10F1	VNMI	indirect for nmi vector
10F2	NMIFLG	nmi in progress flag
10F3	AUTOFG	automatic drive initialization flag
10F4-10FF		unused ram
1100	BUFS	start of data buffers
1100-1FFF		data buffer # 0
1200-12FF		data buffer # 1
1300-13FF		data buffer # 2
1400-1CFF		unconnected
1D00-1FFF	FBUFS	format download area
2000-20FF		data buffer # 3
2100-21FF		data buffer # 4
2200-22FF		data buffer # 5
2300-23FF		data buffer # 6
2400-2FFF		unconnected
3000-30FF		data buffer # 7
3100-31FF		data buffer # 8
3200-32FF		data buffer # 9
3300-33FF		data buffer # 10
3400-3FFF		unconnected
4000-40FF		data buffer # 11
4100-41B3	BAM0	bam drive zero
41B4-41FF	NAMBUF	directory buffer
4200-42B3	BAM1	bam drive one
42B4-42FF		not used
4300-433A	CMDBUF	command buffer
433A	STRSIZ	string size in command buffer
433B	TEMPSA	temporary secondary address
433C	CMD	temporary job command
433D	LSTSEC	last sector
433E-433F	BUFUSE	represents available buffers for channels bit = 1 indicates used buffer
4340-4341	DSKID	current disk id - drive 0
4342-4343		current disk id - drive 1
4344	SECINC	sector increment for sequential files
4345	ENTFND	directory entry found flag
4346	DIRLST	directory listing flag
4347	CMDWAT	command waiting flag
4348	LINUSE	represents available logical indexes. bit = 1 indicates free LINDX. command channel & error channel use 7 & 6
4349	LBUSED	last buffer used
434A	ERBLKS	number of blocks before abort
434B	REC	record size
434C	TRKSS	track of side sector
434D	SECSS	sector of side sector
434E-435B	LSTJOB	15 entries, 1 byte each. last job entered in que. used to retry last job and to extract drive # last used.
435C	REVCNT	error recovery count. set at 10 attempts
425D-436A	ERRCNT	15 entries, 1 byte each. error count on job. each job attempted 10 times before a hard error generated
436B-4372	DIRENT	8 entries, 1 byte each. contains directory entry of. file associated with channel
4373	ERWORD	error word for recovery
4374	PRGSEC	last program sector
4375	WLINDX	write logical index
4376	RLINDX	read logical index
4377	NBTEMP	number of blocks temporary
4379	CMDSIZ	length of command string + 1
437A	CMDNUM	command number
437B	CHAR	character under parse:
437C	LIMIT	pointer limit in compar
437D	F1CNT	file stream 1 count
437E	F2CNT	file stream 2 count
437F	F2PTR	file stream 3 count
4380-4385	FILTBL	table of filename positions in CMDBUF. 5 entries, 1 byte each, therefore, 5 filenames max in cmd string. corresponding entries point at drive number for filename, if present, otherwise first char of filename. if d# present, pointer is moved up to 1st char of filename after d# is set in HDRS
4386-438A	FILTRK	track of 1st block in file during searches. bit7 = 1 indicates pattern matching
438B-438F	FILSEC	sector of 1st block in file searches.
4390	PATFLG	pattern presence flag
4391	IMAGE	file stream image
4392	DRVCNT	number of drive searches
4393	DRVFLG	drive search flag
4394	LSTDRV	last drive without error
4395	FOUND	found flag in directory searches
4396	DIRSEC	directory sector
4397	DELSEC	sector of 1st available entry
4398	DELIND	index of 1st available entry
4399	LSTBUF	= 0 if last block
439A	INDEX	current index in buffer
439B	FILCNT	counter, file entries
439C	TYPFLG	match by type flag
439D	MODE	active file mode (r, w)
439E	JOBRTN	job return flag
439F-43DB		unused
43DC-43FF	ERRBUF	error message buffer
4400-CFFF		unconnected

4040 Dual Disk ROM Map

Loc.	Label	Description
D000	CODE	controller format code
D2A1	CMDTBL	command search table .byt 'ivdmbupcrsn' (initialize, verify-dir, duplicate, m-, b-, user, position, copy, rename, scratch, new)
D2AC	CJUMPL	command jump table low bytes .byt $CA : INTDRV .byt $F3 : VERDIR .byt $50 : DUPLCT .byt $AF : MEM .byt $B6 : BLOCK .byt $0F : USER .byt $EA : RECORD .byt $54 : DSKCPY .byt $7C : RENAME .byt $C1 : SCRTCH .byt $17 : NEW
D2B7	CJUMPH	command jump table high bytes .byt $EC : INTDRV .byt $E6 : VERDIR .byt $E3 : DUPLCT .byt $E7 : MEM .byt $E8 : BLOCK .byt $E8 : USER .byt $FC : RECORD .byt $E4 : DSKCPY .byt $E6 : RENAME .byt $E2 : SCRTCH .byt $E2 : NEW
D2BB	STRUCT	structure images for commands .byt %01010001 : DSKCPY .byt %11011101 : RENAME .byt %00011100 : SCRATCH .byt %10011110 : NEW .byt %00011100 : LOAD
D2C7	TRKTBL	track/group table .byt 17,24,30,35
D2CB	MODLST	mode table .byt 'rwam'
D2CF	TPLST	file type table .byt 'dspul'
D2D4	TYPLST	1st character in name of file type .byt 'dspur'
D2D9	TP1LST	2nd character in name of file type.byt 'eerse'
D2DE	TP2LST	3rd character in name of file type .byt 'lqgrl'
D2E3	ER00	error flag variables for bit: .byt 0
D2E4	ER0	.byt $3F
D2E5	ER1	.byt $7F
D2E6	ER2	.byt $BF
D2E7	ER3	.byt $FF
D2E8	IPBM	.byt $41,$42
D2EA	SECTRK	sectors per track table .byt 17,18,19,21, 9, 2, fm2040 .byt 14,15,16,18,28,30, fm2030
D2F8	TABJMP	controller : sei, jump to wait loop
D301	PEZRO	error display routine. blinks the error # + 1 in all three leds
D32B	DSKINT	initialize disk for PU10 : power up diagnostics
D348	PU10	fill zero page accending pattern
D34E	PU20	then test zero page
D362	RM10	test two 64k-bit roms ; enter .x = start page, exit if ok
D3A0	CR20	test all common ram except page $1000
D3DC	DIAGOK	diagnostics ok so far : test controller
D3F4	INTTAB	initialize buffer pointer table
D46B	SETSEC	set up sector/track table depending on controller used
D47A	SETS20	controller error
D47F	SETS30	set up sectors/track in ram
D48D	SETERR	set up power on message 'cbm dos v2.1'
D492	PONBMP	final set up to start
D4A7	IDLE	idle loop : does housekeeping while waiting for job
D50B	ATNIRQ	atn irq process : irq on atn, listen to pet, clear stack
D54A	DCDE	decide : talk, listen, secondary address, other
D5D0	LISTEN	set listen routine : main routine
D65C	LSTRTN	listen routine
D660	TALK	set talk routine : main routine
D66B	NOTLK	from TALK : no talk - rts
D69C	TLKRTN	talk routine
D6B0	NXTTS	returns next available track and sector given current t & s allocation is from track 18 towards 1 & 35 by full tracks
D6E7	NXTERR	from NXTTS : disk full error
D6FE	FNDNXT	find the next optimum sector
D747	INTTS	returns optimum initial track, sector
D76C	FNDSEC	from INTTS : find sector
D789	SETBMP	set (indirect) bam pointer by DRVNUM
D795	AVAIL	load track bam into TEMP and finds available sector in track
D7BD	AVCK	bit map validity check
D7DB	MAXSEC	returns # of sectors located on specific track : .a = track #
D7E7	TRKNUM	from MAXSEC : track number table .byt 36,31,25,18
D7EB	ERRTAB	error message table : leading error numbers. text with 1st and last characters or'ed with $80. tokens for key words are less than $10 (and'ed with $80)
D8E4	MOVERR	recursive (2) error message routine
D925	ERROR	controller error entry point (.a = error #, .x = job #)
D95C	CMDER2	command error : display error message
D95F	CMDER3	from CMDER2 : clear CMDBUF, set err leds, free internal channel, clear pointers, purge stack
D98C	TLKERR	talker error recovery : if command channel, release DAV. if data channel. force not ready and release channel.
D999	LSNERR	listener error recovery : if command channel, release RFD, if data channel, force not ready and release channel.
D9B1	HEXDEC	convert hex to bcd
D9C1	BCDDEC	convert bcd to ascii dec. return bcd in .x, store ascii in (temp).y
D9D2	OKERR	transfer error message to error buffer
DA1C	FRETS	mark a track, sector as free in bam
DA35	SETLDS	turn on activity led specified by drive number
DA48	ERROFF	turn off error led specified by drive number
DA54	STDIR	directory loading function. get the buffer and get it started.
DB0C	MOVBUF	transfer filename to listing buffer
DB1A	GETDIR	get character for directory loading
DB34	NUMFRE	calculate number of free blocks on drive number
DB5B	PARSXQ	parse and execute string in command buffer
DB9F	ENDCMD	successful command termination
DBA9	SCREND	from ENDCMD : scratch entry
DBBE	CLRCB	clear command buffer
DBC9	CMDERR	command level error processing
DBD2	SIMPRS	simple parser
DBE6	PRSCLN	parse colon
DBEF	TAGCMD	tag command string : set up command structure image and file stream pointers
DBF4	TC25	from TAGCMD : no file error
DC61	TC80	from TAGCMD : bad syntax error
DC69	PARSE	parse string : looks for special characters returning when variable character is found
DCB6	CMDSET	command set : initialize command tables, pointers, etc.
DCDF	CMDRST	command reset : clear variables, tables
DD10	ONEDRV	set 1st drive and table pointers
DD1E	ALLDRS	set up all drives from F2CNT
DD3A	SETDRV	set drive number : determines drive # from text or uses default .a:in: index; out: cmdbuf .y:in: default drive; out: drive #. - if default
DD64	SETANY	set drive from any configuration
DD8C	TOGDRV	toggle drive number
DD95	FS1SET	set pointers to one file stream and check type
DDBB	TST0V1	test character in accumulator for " 0 " or " 1 "
DDC8	AUTOIT	rsr test subroutines : this auto initialization subroutine will check if drive # is initialized. if catalog calls this routine before any header info is transferred, this routine works. this routine will end in error if any error but disk id occurs
DE10	OPTSCH	optimal search for lookup and find file
DE6B	SCHTBL	search table .byt 0,$80,$41, 1, 1, 1, 1 .byt $81,$81,$81,$81,$42,$42,$42,$42
DE7A	LOOKUP	look up all files in stream and fill tables with info
DE8D	LK15	from LOOKUP : toggle drive number
DEB7	FFRE	find next filename matching any file in stream and return with entry found stuffed into tables
DEE1	FNDFIL	from FFRE : find file continuous re-entry, no channel active
DF04	COMPAR	compare all filenames in stream table with each valid entry in the directory
DFB9	CMPCHK	check table of unfound files
DFDA	SRCHST	search directory : returns with valid entry eith delind = 0 or returns with 1st deleted entry with delind = 1
DFDA	SRCHST	initiate a search
E043	SEARCH	continue a search
E069	TRNAME	transfer filename from command buffer .a:string size .x:starting index in command buffer .y:buffer number
E083	TRCMBF	transfer command buffer to other buffer .x starting index in command buffer .y:buffer number
E0A1	FNDLMT	find the limit string in command buffer: pointed to by .x
E0C9	GETNAM	get file entry from directory : called by STDIR and GETDIR
E0D9	GNSUB	from GETNAM : get name subroutine
E1B4	NEWDIR	new directory in listing
E1FA	MSGFRE	display 'blocks free' message in directory buffer
E20B	FREMSG	.byt 'blocks free.'
E217	NEW	new (format) a diskette
E2C1	SCRTCH	scratch file(s)
E31D	DELFIL	delete file by links
E345	DELDIR	delete directory entry
E350	DUPLCT	duplicate diskette
E399	CPYD1	copy blocks from one drive to the other
E3B3	CPYTRK	copy one track
E3DC	READS	read temp + 2 blocks in
E400	WRITES	write temp + 2 buffers out
E420	FORMAT	transfer format code to bufs 1 + 2 and start controller formatting
E454	DSKCPY	check for type and parse special case

E476	DX0000	from DSKCPY : normal parse
E4A8	PRSEQ	from DSKCPY : special case - parse SEQ
E4AF	X0015	from PRSEQ : bad syntax error
E4CF	CPYDTD	copy disk to disk routines
E4F8	EXLP0	from CPYDTD : pull needed variables from stack
E523	FIXIT	from CPYDTD : push needed variables onto stack
E561	TRFNME	transfer name from directory buffer to command buffer
E58E	COPY	copy file(s) to one file : concat
E5B8	COP01	from COPY : file type mismatch error
E5DA	CY	from COPY : check files for existance
E61E	OPIRFL	open and set up internal read file
E65E	GIBYTE	get in a byte
E67C	RENAME	rename file name in directory
E6C3	CHKIN	from CHKIO
E6E0	CHKIO	check i/o file for existance - entrance point
E6F3	VERDIR	validate files with bam. create new bam according to contents of files entered in directory
E6F3	VALDAT	same as VERDIR
E74B	MRKBAM	mark bam with file sectors : called by VERDIR
E773	NEWMPV	set new bam : called by VERDIR
E776	NEWMAP	from NEWMPV : build a new BAM on diskette
E7AE	ECHKSM	E rom checksum .byt 0
E7AF	MEM	memory access commands
E7D1	MEMEX	(m-e) memory execute
E7D4	MEMRD	(m-r) memory read
E7FE	MEMERR	memory command error
E803	MEMWRT	(m-w) memory write
E80F	USER	user access commands
E816	USRINT	'u0' resets usrjmp vector to point to $FFEA
E81F	US10	execute code by usrjmp table : use USREXC to determine action
E825	USREXC	from US10 : determine user action required and proceed
E837	OPNBLK	open direct access buffer from available buffer *
E845	OB05	from OPNBLK : no channel available error
E8B6	BLOCK	block commands
E8C1	BLK10	from BLOCK : bad block command error
E8C6	BLK30	from BLOCK : bad syntax error
E8CB	BLK40	from BLOCK : find command
E8F8	BCTAB	block command table .byt 'afrwep'
E8FE	BCJMP	block command jump table (as follows)
		block-allocate (b-a) $E999
		block-free (b-f) $E990
		block-read (b-r) $E9FC
		block-write (b-w) $EA19
		block-execute (b-e) $EA49
		block-pointer (b-p) $EA60
E90A	BLKPAR	parse block parameters
E93C	ASCHEX	convert ascii to hex and store conversion in tables
		.y: pointer into command buffer
E98D	DECTAB	decimal table .byt 1,10,100
E990	BLKFRE	(b-f) block-free
E999	BLKALC	(b-a) block-allocate
E9D1	BA40	from BLKALC : no block error
E9DC	BLKRD2	start of block-read subroutines
E9E8	BLKRD3	from BLKRD
E9FC	BLKRD	(b-r) block-read
EA05	UBLKRD	user direct block read : last char = $FF
EA19	BLKWT	(b-w) block-write
EA3D	UBLKWT	user direct block write : no last char
EA49	BLKEXC	(b-e) block-execute : read block and execute
EA60	BLKPTR	(b-p) buffer-pointer
EA75	BUFTST	test for allocated block related to secondary address
EA83	BT15	from BUFTST : no channel error
EA95	BKOTST	test block operation parameters
EA98	BLKTST	test for legal block and set up drive, track, and sector
EAB3	FNDREL	find relative file
		inputs : recl - 1 byte = lo record *
		rech - 1 byte = hi record *
		rs - 1 byte = record size
		recptr - 1 byte = 1st byte wanted from record
		outputs : ssnum - 1 byte = side sector number
		ssind - 1 byte = index into side sector
		relptr - 1 byte = ptr to first byte wanted
EAD1	MULPLY	multiply : result = rec. * x rec. size + rec. ptr.
EB13	DIV254	divide : result = quotient, remainder = accum + 1
EB13	DIV254	divide by 254
EB16	DIV120	divide by 120
EB1C	DIV100	main division routine
EB2E	DIV200	divide by 256
EB7E	ZERRES	zero result
EB87	ACCX4	multiply accum x 4
EB8A	ACCX2	multiply accum x 2
EB92	ADDRES	add accum to result : result = result + accum + 1,2,3
EB9F	USEDTS	mark track, sector, (BMPNT) as used
EBB4	FREUSE	calculate index into bam for FRETS and USEDTS
EBCE	BMASK	bit mask table .byt 1,2,4,8,16,32,64,128
EBD6	DBLBUF	toggle active buffer * in BUFNUM
EBEB	PIBYTE	write to channel : alternate entrance point
EBFD	PUT	write to channel : main entrance point
EC1E	L42	from PUT and PIBYTE : write to command channel
EC37	TSTJOB	test if job(.x) is done yet, if not done then return, if ok then return else redo it
EC4A	RECOV	recover job : bump head to track 1 and try again
EC5B	REC1	from RECOV : test REVCNT for * times for recovery, set up
EC7D	OK	from TSTJOB : c = 0, everything ok, return
EC7F	AGAIN	from TSTJOB : store LSTJOB back on JOBS to try again
EC85	NOTYET	from TSTJOB : c = 1, job not finished yet, try again
EC87	WATJOB	wait until job(.x) is done the return
EC94	SETHDR	set header of active buffer of the current lindx to trk, sec, ID
ECB6	PUTBYT	put accum into active buffer of lindx. if no active buffer, file not open error generated
ECC2	PUTB1	from PUTBYT : actual accum into buffer routine
ECCA	INTDRV	initialize drives (command)
ECE4	INITSU	initialize drive (DRVNUM) · BUMP head to trk 1, setup for trk 18, sector 0 for job SEEK to get BAM, disk ID
ECFF	INITDR	from INTDRV : actual initialization routine
ED22	STRDBL	start read double buffering, use track, sector as starting block
ED46	RDBUF	from STRDBL : set up for READ job on track, sector
ED4A	WRTBUF	set up for WRITE job on track, sector
ED4C	STRTIT	start READ/WRITE job
ED6E	FNDRCH	find read channel
ED89	FNDWCH	find write channel
EDA6	TYPFIL	get file type
EDB0	GETPRE	from GETBYT : get active buffer number, lindx, rts
EDB8	GETBYT	read byte from active buffer and set flag if last data byte, if last then z = 1 else z = 0
EDD7	RDBYT	read a character from file and read next block of file if needed, set CHNRDY = EOI if end of file
EE1E	WRTBYT	write character to channel and write buffer to disk if it's full
EE24	WRT0	from WRTBYT . write buffer to diskette
EE47	INCPNT	increment pointer of active buffer by accum
EE47	INCPTR	same as INCPNT : commodore patch
EE54	SETDRN	set DRVNUM to drive indicated by LSTJOB of active buffer
EE60	GETWCH	sets up buffer * and allocates lindx : .a = * buffers needed
EE60	GETWCH	get write channel . carry set for write
EE63	GETRCH	get read channel : carry clear for read
EE64	GETR2	from GETWCH : main routine to set up buffer *
EE8C	GBERR	from GETR2 . no channel error
EEA4	FRECHN	free channel associated with secondary address. free read and write channels. don't free channel 15
EEAB	FRECO	from FRECHN : actual free channel routine
EEAB	FRERD	same as FRECO : commodore patch
EEAB	FREWRT	same as FRECO . commodore patch
EEBE	RELINX	release lindx
EECF	RELBUF	given secondary addr, free its read channel. release bufs (lindx)
EF03	GETBUF	get a free buffer number
EF34	FREBUF	allocate a buffer number
EF48	CLRCHN	clear all channels : sa = 1-14
EF54	CLDCHN	channels cleared
EF79	FNDLNX	find a free lindx to use. mark as used in LINUSE
EF95	GBYTE	get the next character from a channel
EFA3	GET	from GBYTE : actual get routine
EFD7	RNDGET	direct file character get
EFF9	SEQGET	SEQ file character get
F001	GET6	check if directory load. if not branch to SEQGET
F00C	GETERC	error channel character get
F044	NXTBUF	read next buffer of a file. follow links in first two bytes end of file if 1st byte = 0, 2nd character = length
F057	DRTRD	entrance point for direct block-read
F05B	DRTWRT	entrance point for direct block-write
F05D	DRT	direct block read/write routine
F06C	OPNIRD	open internal read channel (secondary address = 17)
F07C	OPNIWR	open internal write channel (secondary address = 18)
F083	NXDRBK	allocate next directory block on 18 and mark as used in bam
F0C1	SETPNT	set pointer. .a = new pointer value
F0D3	FREICH	free the internal read (sa = 17) and write (sa = 18) channels
F0E1	GETPNT	read the active buffer pointer
F0EF	DRDBYT	direct read byte, .a = byte * to read
F0FF	BUFIND	index table containing high byte addresses of buffers
		.byt $11,$12,$13
		.byt $20,$21,$22,$23
		.byt $30,$31,$32,$33
		.byt $40,$41,$42,$43
F10E	SETLJB	use last job for drive num, command is used for job
F116	SETJOB	set job up and check trk, sec : .a = command for job, .x = job *
F155	TSERR	illegal track and sector error
F15D	HED2TS	from TSERR : set up trk and sec for error
F16E	TSCHK	track and sector checkout routine
F180	VNERR	write to wrong version error
F19D	DOIT	do job in .a, set up error count and LSTJOB. return when job complete. jump to error if error returns
F1A9	ADDFIL	add file to directory
F279	OPEN	open channel from ieee. parses the input string that is sent as an open data channel, load, or save. channels are allocated and the directory is searched for the filename contained in the string
F294	OP02	from OPEN : load last program
F2B5	OP021	from OPEN : load directory
F2C1	OP04	from OPEN : open directory as sequential file
F2D6	OP041	from OPEN : open '*' direct access file
F2DD	OP0415	from OPEN : program file type
F2F8	OP05	from OPEN : 'syntax error' generated
F39F	OP81	from OPEN : check for replace ('@')
F3AE	OP815	from OPEN : 'bad filename' error generated
F3B3	OP82	from OPEN : save/write with replace ('@')
F3F9	OP90	from OPEN : open read and load
F400	OP95	from OPEN : 'file not found' error generated
F420	OP115	from OPEN : 'file type mismatch' error generated
F425	OP120	from OPEN : everything a-ok! - continue with process
F45B	OPREAD	from OPEN : open a read file
F49B	OPWRIT	from OPEN : open a write file
F4A7	OPFIN	from OPEN : procedure finished
F4BF	CKTM	check mode or file type
F4C7	CKM1	from CKTM : check mode
F4D4	CKT1	from CKTM : check type
F4DF	APPEND	append a file
F509	LOADIR	load directory
F52B	LD01	from LOADIR : load by name
F536	LD02	from LOADIR . load one directory
F58D	CLOSE	close file associated with secondary address
F59C	CLS10	from CLOSE : close directory file
F5AC	CLSALL	from CLOSE : close all files
F5BA	CLSCHN	from CLOSE : locate and close specific file type
F5E3	CLSREL	from CLOSE : close relative file
F612	CLSWRT	from CLOSE : close a write channel
F655	MAPOUT	write out the bit map to the drive in LSTJOB (active)
F671	MAPCHK	verify that the bam block count matches the bits
F6A4	CLSDIR	directory close on open write file
F747	OPNRCH	open a read channel with 2 buffers. will insert sa in lindx and initializes all pointers. if relative, ss and pointers are set
F7A8	OR30	from OPNRCH : sequential file set up
F7BA	INTPNT	initialize variables for open channel, LSTJOB, sets active buffer number, LSTCHR, buffer pointers in BUFTAB = 2
F7E6	OPNWCH	open a write channel with 2 buffers
F895	PUTSS	put byte into side sector
F89D	SCFLG	set/clear flags
F89F	SETFLG	set flag entrance point
F8A5	CLRFLG	clear flag entrance point
F8AE	TSTFLG	test for state of flag
F8B3	TSTWRT	test for write
F8BF	TSTCHN	test for active files from lindx table
		c = 1 file not active : .x = 18, .y = ?, .a = ?
		c = 0 file active : .x = endlnd, .y = lindx, .a = ?
F8F1	SCRUB	write out buffer if dirty
F8FD	SETLNK	put track, sector into buffer
F90C	GETLNK	get link from buffer into track and sector
F919	NULLNK	set track link = 0 and sector link = last non-zero character
F92B	SET00	set up pointer to buffer
F93B	CURBLK	read track and sector from header
F93E	GETHDR	from CURBLK : get header
F952	WRTAB	set up for write in job que, branch to SJ10
F959	RDAB	set up for read in job que, branch to SJ10
F960	WRTOUT	set up for write in job que, branch to SJ20
F967	RDIN	set up for read in job que, branch to SJ20
F96E	WRTSS	set up for write in job que, branch to RDSS
F975	RDSS	set up for read in job que
F977	RDS5	accessed by WRTSS
F981	SJ10	accessed by WRTAB and RDAB
F98D	SJ20	accessed by WRTOUT and RDIN
F997	RDLNK	set track/sector from link in buffer
F9A7	B0TOB0	transfer bytes from one buffer to another
		registers in : .a = number of bytes
		.y = source buffer number
		.x = destination buffer number
F9C3	CLRBUF	clear buffer given
		register in : .a = buffer number
		registers out : .y, .a = 0
F9D4	SSSET	set side sector pointer to 0, register out: .a = side sector *
F9DE	SSDIR	set DIRBUF with current side sector pointer
		register in : .a = low byte
F9EB	SETSSP	set DIRBUF and BUFTAB with current side sector pointer
		register in : .a = low byte
F9FA	SSPOS	position side sector and BUFTAB to ssnum & ssind
		flag: .v = 0 ok : .v = 1 out of range
FA1D	IBRD	indirect block read
		register in : .a = buffer * for read / .x = lindx
		(dirbuf).y points to track, sector to be read
FA23	IBWT	indirect block write
		register in : .a = buffer * for write / .x = lindx
		(dirbuf).y points to track, sector for write
FA27	IBOP	code for IBRD and IBWT routines
FA47	GSSPNT	get side sector pointer
FA4E	SCAL1	calculate * side sector blocks required
FA53	SSCALC	from SCAL1
FA5E	ADDT12	add * side sectors needed x 120
FA68	SSTEST	test ssnum and ssind for residence and range.
		variables : ssnum, ssind, dirbuf
		flags : .n range, .v residence error
		0 ok 0 yes er0
		0 maybe 1 no er1
		1 no 0 yes er2
		1 no 1 no er3
FA95	GETACT	get active buffer number
		variables : buf0, buf1, lindx
		registers out : .a = active buf *, .x = lindx, .n = 1 no active buf
FAA0	GAFLGS	get active buffer *, set LBUSED and flags
		registers out : .a = active buf *, .x = lindx
		flags : .n = 1:no active buf, .v = 1:dirty flag
FAB9	NXTREC	mark end of record then move on to next record
FB25	NRBUF	read track, sector link into buffer
FB65	RELPUT	write relative data into buffer
FB94	WRTREL	write relative buffer
FBD9	CLREC	put zeros into balance of relative record
FBEB	SDIRTY	set dirty flags
FBF6	CDIRTY	clear dirty flags
FC01	RDREL	read relative file
FC53	SETLST	set last character in record
FC95	FNDLST	find last character in record
FCAE	SSEND	position side sector and BUFTAB to end of last record
FCE5	BREAK	illegal system track or sector error encountered
FCEA	RECORD	position relative pointers to given record number or last record if out of range
FD58	POSITN	position relative data block into active buffer and next block into inactive buffer
FD7A	POSBUF	position proper data blocks into buffers
FDB8	BHERE	check if required block is in buffer
FDCA	NULBUF	set null records in active buffer for extension
		variables : NR, RS, LX, act-buf
		in : NR = last record position in previous buffer
		out : NR = last record pos in buf for next NULBUF or to set LSTCHR
FDEC	ADDNR	add next record to record size and leave in accumulator. if c = 1 then a buffer boundary has been crossed
FE04	ADDREL	add blocks to relative file
FE48	AR20	from ADDREL : too many ss's 'file to large' error generated
FE4D	AR25	from ADDREL : calc * blocks needed and check against avail
FF33	NEWSS	generate new side sector and fix old side sectors to reflect it
FFE1	NMI	non maskable interrupt : jmp ($10F0)
FFE4	PATCH	a commodore patch to boost to $FFE9
FFE9	FCHKSM	F ROM checksum .byt 0
FFEA	UBLOCK	default jump table for user commands
		UBLKRD user block read (u1) : $EA05
		UBLKWT user block write (u2) : $EA3D
		user jmp through (u3) : $1300
		user jmp through (u4) : $1303
		user jmp through (u5) : $1306
		user jmp through (u6) : $1309
		user jmp through (u7) : $130C
		user jmp through (u8) : $130F
FFFA		kernal nmi : $FFE1
FFFC		kernal disk initialization : $D32B
FFFE		kernal atn irq process : $D50B

4040 Dual Disk Controller RAM Usage

The 6530 Disk Controller contains 64 bytes of RAM for use by the 6504 CPU: 0000-001F is used for storage
0020-003F is the stack seen by the 6504 at 0100-013F

Loc.	Label	Description
0000	CLOCK	controllers clock
0001 -0002	MTRTM	motor timer : drive 0 / drive 1
		(+) when motor fully on
		(0) when motor should be turned off
0003 -0004	DRVST	drive status words bits 0-5 track #
		bit 6 stepping 0 = no, 1 = yes
		bit 7 accelerating 0 = no, 1 = yes
0005 -0006	STEPS	number of steps to new track
0007	COW	used with interrupt
0008 -0009	WORK	(+0) closest seek distance
		(+1) closest seek direction
000A	DTRCK	number of spaces for format
000B	DSECT	number of sectors until desired sector
000C	CSECT	closest sector from current position
000D -0011	STAB	sector header table : same format as HDRS table
0012	DRIVE	current drive # trk # # sec freq
		1-17 21 FE
		18-24 19 FC
		25-30 18 DE
		31-35 17 DC
0013	TRACK	track number for closest seek
		bits 0-1 part of id
		bits 2-7 track number
0014	NEXTS	next sector on drive
0015	SECTR	number of sectors/track
0016 -0017	BUFPT	lo/hi pointer into BUFS table
0018 -0019	HDRPNT	lo/hi pointer into HDRS table, if $FF then no job
001A	FTNUM	format count : $FF = no action
001B -001C	IP	(* indirect pointer *)
001D	CNT	error count
001E	JOB	current job being done
001F	JOBNUM	current job id
0020 -003F		stack for 6504
0040	VIAA	**MOS 6522** $0040-004F
0040	VB	port b
		bits 0-1 stepper motor drive #1
		bits 2-3 stepper motor drive #0
		bit 4 motor 1 off
		bit 5 motor 0 off
		bit 6-7 unused
0041	DIN	port a : data input
0042	VDDRB	data direction register b
0043		appears unused by FDC
0044	TILL	timer 1 latch and counter low
0045	TIMER	timer 1 counter high
0046 -004A		appears unused by FDC
004B	ACR	auxillary control register
004C	PCR	peripheral control register
		bit 0 set to 0
		ca1: byte ready 1 = yes, 0 = no
		bits 1-3 ca2 : fill/sync
		normal : xc
		sync/fill: xe
		bit 4 set to 1
		cb1: error detected 1 = yes, 0 = no
		bits 5-7 cb2 : read/write
		write : dx
		read : fx
004D	IFR	int flag register
004E	IER	int enable register
0080	MITA	**MOS 6530** $0080-008F
0080	DOUT	port a : data out
0081	EOUT	direction port a
0082	PB	port b
		bit 0 switch 0 = drive #0
		1 = drive #1
		bits 1-2 frequency (bit density)
		bit 3 write protect 1 = yes
		bit 6 sync detect 1 = no, 0 = yes
0083	DDRB	data direction register b
0084 -008E		appears as unused by FDC
008F	MITAT	timer/1024
		Common RAM 6404: $0400-04FF
0400	TICK	interrupt interval 6502: $1000-10FF
0401	DELAY	motor acceleration delay
0402	CUTMT	motor cutoff time
0403 -0411	JOBS	job que
		bit 7 0 = ignore, 1 = job present
		bits 6-4 mode
		000: read (8) (0) :read data block
		001: write (9) (1) :write data block
		010: verify (A) (2) :verify data block written
		011: seek (B) (3) :seek specific track and sector
		100: bump (C) (4) :restore placement of head: trk 1
		101: jump (D) (5) :jump to buffer code
		110: execute (E) (6) :start motor then jump
		bit 0 :drive: 0:#B, 1:#A
0421 -0498	HDRS	headers of current blocks : 15*8
		-3: sync 1: id2
		-2: sync 2: track # (bits 7-6 part of id)
		-1: "08" 3: sector # 5: off
		0: id1 4: checksum 6, 7: spare
0499 -049C	TAB1	# sectors/track initialized by dos
049D	GAP1	gap 1 size set by dos
049E	GAP2	gap 2 size set by dos: used in format for min # of bytes
049F	VERNUM	dos version number
04A0	ACTJOB	active job number
		data on diskette preceded by: sync, sync, "07"
		chksum follows 256 data bytes then 16 spacing bytes
0500 -13FF	BUFS	set of 15 1-block (256 word) buffers

4040 Dual Disk Controller ROM Map

The 6530 Controller contains 1K of ROM. The following map is actually for the 2040 (DOS 1.0) drive, but the 2040 and 4040 Controllers are virtually identical. 8050 Controller ROM Map not available at this time.

Loc.	Label	Description
0500	FORMT	format code – mode 101 (d)
0504		initialize head phase and track number
0538	L216	initialize track number and move head to desired track
053E	L213	formatting in progress – check if correct track : bne L216
0548	L217	head is on desired track : init sec, disable cb1 flag, check wpsw
0561	L299	compute header checksum
0572		set up for writing 0's to blank out diskette
0577	L301	write 3*256 bytes : 3 blocks of 0's
0581	L377	write initialized data block : sync, chksum, sync, header, etc.
05B0		set up for spacing 16 bytes between header
05B2	L304	loop to space 16 bytes between header
05B8		increment sector number and check if last one : beq L378
05C7		update checksum quickly, then jmp L377
05D1	L378	test if bump into sync character after 256 bytes
		branch if no sync after 250 words to L291 for more testing
		if too small error : branch to DERR
		otherwise, branch to L293, keep on going
05E1	L291	test 41 more characters for sync : branch to L294 if found
05E8	L292	check if too big error, beq DERR
05F0	L293	make spacing larger : jmp L217
05F3	L294	increment track number : check if format error, bne FV1
05FB	DERR	reset FTNUM, set up format error code, jmp ERROR
0604	FV1	continue
060E	LOOP	search for specific block, inc + check if last track, beq L219
063A	L219	format is finished
FC00	JOHN	initialization
		initialize stack (S = $3F), CLD, VDDRB = $FF (all output)
		CUTMT = $FF DDRB = $07 FTNUM = $FF PCR = $FC
		VB = $FF IER = %10010010 ACR = %1 TILL = 0
		BUFPT = 0 FMTFLG = 0 all JOBS = 0 all STEPS = 0
		TICK = #15 MITAT = #15 (irq every 15.36 ms)
		DRVST = $80 DRVST + 1 = $80 (set motor as still)
		DELAY = $50 HDRPT + 1 = #>HDRS
FC47		loop until job found, turn on motors if needed
		.x = drive #, .y = job #
FC47	START	ldy #15-1 : load # jobs
FC49	L010	check if valid job, if so, which drive
FC55		test motor status, turn on if not and set time for accel delay
FC6A	L012	test motor speed
FC74	L013	test head status, if not moving branch to QUE
FC77	L014	scan next job, if next job, branch to L010
FC7A	END	branch to START
FC7C		motor is on and head is still, if head is on right track, start
		processing by branching to GOTU, otherwise, move to closest
		needed track : .x = drive #, .y = job #
FC7C	QUE	initialize to maximum distance + 1, and set .y for max job #
FC83	L020	init JOBS and JOBID by .y offset
FC90		test if on right track
FC9E		find closest seek
FCB1	L022	decrement .y loop for all jobs
FCB4		set up seek to closest track
FCCE	FIN	loop to search table again
FCD0	TAB1	sectors/track table : .byt 17,18,20,21
FCD4	ANDA	.byt $F3
FCD5	TAB3	TAB3 + 1 = tracks # : .byt $FC,31,25,18
FCD9		head is on desired trk, set drive switch, # sectors, & bit density
FCD9	GOTU	check if motor to speed, branch to FIN if not
FCDD		set up for check of track zone
FCE2	L032	check for track zone
FCE8	L031	set # sectors/track by results of L032
FD02	EXE	**Job Routine:** execute : mode = 110 (E)
		check if execute, branch to EX if so, check if bump, branch to bump if so; jmp seek
FD0D	EX	execute routine : get job #, calc buffer address, branch to it
FD16	BUMP	**Job Routine:** bump the hub : mode 100 (C)
		load drive #, accel to track, isolate drive, set head to phase "a"
		set head to max distance (#256-116), jmp DONE
FD2D	WSECT	decide which sector to service
FD43	L480	check which job type, check track, drive
FD8D	HPNT	adjust header pointer : job*8 + hi byte of HDRS into HDRPNT
FDA1	FSNUM	fix sector number for fake seek
FDAA	READ	**Job Routine:** read a block : mode = 000 (8)
		check if read or write, branch to WRITE is so
FDB1	L100	get the bytes, store in (BUFPT),y, update chksum, jmp DEND
FDC3	DSTRT	start reading data : init chksum, search for header & start of data
FDD6	WRITE	**Job Routine:** write a block : mode 001 (9)
		check if verify, branch to VRFY if so
		check write protect, if ok L198, if no good, ER
FDE4	L198	disable CB1 flag, get correct block
FDEB	L200	write sync mode, load fill code
FDFB	L201	reset port a flag, set 1st sync
FE05	L202	store normal code mode in PCR, set 2nd sync, chksum
FE15	L203	write block, write chksum, change job to verify, end
FE3A	VRFY	**Job Routine:** verify a written data block : mode = 010 (A)
		read data
FE3D	L210	get byte and compare with contents of buffer, add up chksum
FE4E	DEND	end reading data, final chksum compare
FE59	L214	check if decoding error : if not then DONE, else ER
FE61	L212	set for verify error
FE63	ER	branch to error routine ERR
FE65	SEEK	seek to determine next sector number
		init chksum, get block header
FE70	L250	get a byte, store in STAB, update chksum, branch again if more
FE8C		load job # and type, test if seek, branch to ESEEK if so
FEA6	L252	check if id in (HDRPT),y = STAB,y, error if not
		continue loop, at end jmp WSECT
FEB3	ESEEK	**Job Routine:** seek : mode 011 (B)
FEB9	L251	get complete header from STAB,y into (HDRPT),y
FEC1	DONE	set for no error : #1
FEC3	ERR	jmp ERROR : error routine
FEC6	CSERR	lda #9 (chksum error), branch to ERR
FECC	L253	lda #11 (mismatch), branch to ERR
FECE	SRCH	search for specific block
FED2	L412	compute checksum, set up for search for a sector
FEDF	L410	jsr HEAD, set .y for compare : every byte in hdr must be identical
FEE4	L411	compare to header loop, loop entire header
FEF3	HEAD	search for block head : .x = max # trials
FEF9	ERROR	send error code
		# 1: no error
		# 2: can't find block head
		# 3: no sync character
		# 4: data block not present
		# 5: checksum error
		# 7: verify error
		# 8: write with write protect on
		# 9: checksum error in seeked header
		#10: data ran into next header
		#11: disk id mismatch
		#16: decoding error
FF07	ERR1	send job status, make motor stay on longer, check job type
FF1F	L421	purge stack (#$3F)
FF25	L420	get a byte, compare to start of header, branch if not equal to HEAD
FF2D	WATCH	watch for sync characters
		bit PB, bvc L450 ; test if sync present, branch if yes
		bit IFR, bpl WATCH ; test if byte present, loop until yes
		.x contains # of tests, branch to WATCH till .x = 0
		bit PB, rts ; test if sync present then return
FF3F	SYNC	hunt for sync character : set timer for 20 ms limit
FF52	BYTE	get a byte
FF59	OFF	send two bytes, set for normal read mode
FF63	L442	change EOUT and PCR to send sync
		enable CB1, CA1, CB2 (IER = %10011010), get a byte
		reset VIA flags, get next byte, reset VIA, jmp BYTE (get next byte)
FF7C	OUT	byte to be sent is in .x
FF85	IRQH	interrupt for a few milliseconds : set next interrupt, reset timer
FF91	L898	service motor : check if motor on and stepping flag set
FFAB	L941	service stepper motor
FFAD	L921	check if on track, if not then L911
		on track, clear stepping tag, check next stepper (jmp L920)
FFBA	L911	check direction – set out or step in
FFC9		step in (+)
FFD0	L912	step out (-)
FFD8	L913	store new stepper position, test if DRVST ready.
FFE7	L930	pop the stack of .a and .x then rti
FFEB	ANDB	.byt $04,$01
FFED	ANDC	.byt $20,$10
FFEF	ANDD	.byt $0C,$03
FFF1	ANDE	.byt $A0 $50
FFFC		reset and interrupt vectors
FFFC		.word john : $FC00 initialize
FFFE		.word irqh : $FF85 : interrupt

8050 Dual Disk Controller RAM Usage

The 6530 Disk Controller contains 64 bytes of RAM for use by the 6504 CPU: 0000-0029 is used for storage
002A-003F is the stack seen by the 6504 at 0100-013F

Loc.	Label	Description
0000	CLOCK	controller clock
0001	MTRCLK	motor clock : clock/16
0002 -0003	MTRTM	motor timer : drive 0 / drive 1
		(+) when motor fully on
		(0) when motor should be turned off
0004 -0005	DRVST	drive status words
		bits 0-5 track #
		bit 6 stepping 0 = no, 1 = yes
		bit 7 accelerating 0 = no, 1 = yes
0006 -0007	STEPS	number of steps to new track
0008	COW	used with interrupt
0009 -000A	WORK	(+0) closest seek distance
		(+1) closest seek direction
000B	DTRCK	number of spaces for format
000C	DSECT	number of sectors until desired sector
000D	CSECT	closest sector from current position
000E -0012	STAB	sector header table : same format as HDRS table
0013	DRIVE	current drive #
0014	TRACK	track number for closest seek
		bits 0-1 part of id
		bits 2-7 track number
0015	NEXTS	next sector on drive
0016	SECTR	number of sectors/track
0017 -0018	BUFPT	lo/hi pointer into BUFS table
0019 -001A	HDRPNT	lo/hi pointer into HDRS table, if $FF then no job
001B	FTNUM	format count : $FF = no action
001C -001D	IP	(* indirect pointer *)
001E	CNT	error count
001F	JOB	current job being done
0020	JOBNUM	current job id
0021 -0022	DRVTRK	track drive is currently on
0023 -0024	STPCNT	(* step count *)
0025	CHKSUM	(* checksum *)
0026	BI	
0027	FLG2	
0028	NXTJOB	next job : optimal track seek
0029	NXTRK	next track : optimal track seek
002A -003F		stack ram for 6504
0040	VIAA	**MOS 6522** $0040-004F
0040	VB	port b
		bits 0-1 stepper motor drive #1
		bits 2-3 stepper motor drive #0
		bit 4 motor 1 off
		bit 5 motor 0 off
		bit 6 pll control bit
		bit 7 sync detect 1 = no, 0 = yes
0041	DIN	port a : data input
0042	VDDRB	data direction register b
0043		appears unused by FDC
0044	TILL	timer 1 latch and counter low
0045	TIMER	timer 1 counter high
0046 -004A		appears unused by FDC
004B	ACR	auxillary control register
004C	PCR	peripheral control register
		bit 0 set to 0
		ca1: byte ready 1 = yes, 0 = no
		bits 1-3 ca2 : fill/sync
		normal : xc
		sync/fill: xe
		bit 4 set to 1
		cb1: error detected 1 = yes, 0 = no
		bits 5-7 cb2 : read/write
		write : dx
		read : fx
004D	IFR	int flag register
004E	IER	int enable register
0080	MITA	**MOS 6530** $0080-008F
0080	DOUT	port a : data out
0081	EOUT	direction port a
0082	PB	port b
		bit 0 switch 0 = drive #0
		1 = drive #1
		bits 1-2 frequency (bit density)
		bit 3 write protect 1 = yes
		bit 4 odd head select
		bit 6 unused
0083	DDRB	data direction register b
0084 -008E		appears as unused by FDC
008F	MITAT	timer/1024
		Common RAM 6404: $0400-04FF
0400	TICK	interrupt interval 6502: $1000-10FF
0402	CUTMT	motor cutoff time
0403 -0411	JOBS	job que
		bit 7 0 = ignore 1 = job present
		bits 6-4 mode
		000: read (8) (0) :read data block
		001: write (9) (1) :write data block
		010: verify (A) (2) :verify data block written
		011: seek (B) (3) :seek specific track and sector
		100: bump (C) (4) :restore placement of head: trk 1
		101: jump (D) (5) :jump to buffer code
		110: execute (E) (6) :start motor then jump
		011: secsek (B) (3) :X011 100D
		bit 0 drive: 0:#B, 1:#A
0421 -0498	HDRS	headers of current blocks : 15*8
		-3: sync 2: track # (bits 7-6 part of id)
		-2: sync 3: sector #
		-1: "08" 4: checksum
		0: id1 5: off
		1: id2 6, 7: spare
0499 -049C	TAB1	# sectors/track initialized by dos
049D	GAP1	gap 1 size set by dos
049E	GAP2	gap 2 size set by dos: used in format for min # of bytes
049F	VERNUM	dos version number
04A0	ACTJOB	active job number
		data on diskette preceded by: sync, sync, "07"
		chksum follows 256 data bytes then 16 spacing bytes
04A1 -04A2	PHASE	phase of stepper motor
04A3	STPTRK	number of steps per track
04A4	NZONES	number of zones
04A5	SYNDLY	delay on PLLSYN control after SY???? (* off page edge *)
04A6 -04A7	WPSW	write protect switch
04A8 -04A9	LWPT	last state of write protect switch
04AA	PBI	block identifier
04AB	CFLG2	common flag 2
04AC	NSIDES	number of sides on the diskette
04AD	SPDVAR	speed measure
04AE	UNUSED	unused
04AF		
04B0 -04B7	TAB3	track boundary table: up to 4 zones
		data on diskette preceded by : sync, sync, "07"
		chksum follows, 256 data, then approx 16 spacing bytes
04BF	SYNCS	
04C0 -04FF		(* unused by FDC *)

8050 Disk Memory Map

8050 System Constants

Hex Val	Label	Description
$00	NOTRDY	i/o not ready
$00	RDMODE	open read mode
$00	VAL	job code for validate
$01	ATNA	atn active
$01	LISNER	ieee listener flag
$01	RDYLST	i/o ready to listen
$01	SEQTYP	sequential file type
$01	WTMODE	open write mode
$02	APMODE	open append mode
$02	DACO	data accepted - output
$02	DOSVER	dos version
$02	PRGTYP	program file type
$03	MDMODE	open modify mode
$03	USRTYP	usr file type
$04	LOTRK	low track number
$04	NMODES	number of modes within table MODLST ('RWAM')
$04	RELTYP	relative file type
$04	RFDO	ready for data - output
$05	HITRK	high track = lotrk + 1
$05	MXFILS	maximum number of filenames in string
$05	NTYPES	number of file types from TYPLST ('DSPUR')
$06	CMDCHN	command channel = mxchns - 2
$06	NBCMDS	start offset for comparison with table BCTAB ('AFRWEP')
$06	NSSL	number of side sector links
$07	DIRTYP	direct file type
$07	ERRCHN	error channel number = mxchns - 1
$07	ID8050	dos version identifier : 8050
$07	TYPMSK	type mask for matching pattern type
$07	VERERR	controller verify error
$08	EOIO	eoi - output
$08	EOISND	not (eoi) to send
$08	LED1	active led 1
$08	MXCHNS	maximum number of channels
$09	PCMD	commands not parsed error
$0C	LDCMD	load command # / load command image
$0C	MSGLEN	length of 'blocks free' message at $CB29 - FREMSG
$0C	NCMDS	number of commands from CMDTBL ('VIDMBUP&CRSN')
$0D	CR	carriage return
$0F	CMDSA	command channel secondary address
$10	DAVO	data valid - output
$10	ERRSA	error channel secondary address
$10	LED0	active led 0
$10	SSIOFF	offset into side sector for data block pointers
$11	IRSA	internal read secondary address channel
$12	IWSA	internal write secondary address channel
$12	MAXSA	maximum secondary address
$18	DIRLEN	length of directory buffer
$1B	NBSIZ	nambuf text size
$1C	CBPTR	command buffer pointer
$1E	CMDIND	command index * 2
$20	EOII	eoi - input
$20	ERRLED	hardware initialization error led
$20	OVRFLO	overflow flag value
$30	BADSYN	error : general syntax
$31	BADCMD	error : invalid command
$32	LONGLN	error : long line
$33	BADFN	error : invalid filename
$34	NOFILE	error : no file given
$39	NOCFIL	error : command file not found
$3A	CMDLEN	length of command buffer
$3F	LXINT	lindx 0 to 5 free
$3F	UNLSN	ieee unlisten command number
$40	DAVI	data valid - input
$40	DYFILE	dirty file flag
$40	NDACI	no data accepted - input
$41	FM2040	dos format version # for 2040 drive
$42	FM2030	dos format version # for 2030 dnve
$43	FM8050	dos format version # for 8050 dnve
$50	NOREC	error : record not present
$51	RECOVF	error : overflow in record
$52	BIGFIL	error : file too large
$60	FILOPN	error : file open for write
$61	FILNOP	error : file not open
$62	FLNTFD	error : file not found
$63	FLEXST	error : file exists
$64	MISTYP	error : file type mismatch
$65	NOBLK	error : no block
$66	BADTS	error : illegal track or sector
$67	SYSTS	error : illegal system track or sector
$70	NOCHNL	error : no channels available
$71	DIRERR	error : directory error
$72	DSKFUL	error : disk full
$73	CBMV2	'cbm dos v2.5 8050' message number
$74	NODRIV	error : drive not ready
$78	NSSP	number of pointers in side sector
$80	ATNI	atn inactive
$80	EOIOUT	talk with eoi
$80	LRF	last record flag
$80	NRFD1	next record flag for drive 1
$80	READ	controller job type : read
$80	TALKER	ieee talker flag
$81	RNDEOI	random with eoi
$88	RDYTLK	talk no eoi
$89	RNDRDY	random chnrdy = rdytlk + rdylst
$90	WRITE	controller job type : write
$A0	WVERFY	controller job type : write/verify
$B0	SEEK	controller job type : seek
$B8	SECSEK	controller job type : sector seek
$C0	BUMP	controller job type : bump the head
$D0	JUMPC	controller job type : jump to user ml routine
$D9	ERRTOK	size of error message token table
$E0	EXEC	controller job type : execute ml routine

8050 RAM Memory Map with Zero Page Contents at Power Up

Hex Location		Content	CBM Label	Function
00-01	00	EA	USRJMP	User Jump Table Pointer - $FFEA
	01	FF		
02-03	02	00	BMPNT	Bit Map Pointer - $4200
	03	42		
04-09	04	04	TEMP: T0	Temp Work Space
	05	00	: T1	
	06	00	: T2	
	07	05	: T3	
	08	00	: T4	
	09	00		
0A-0B	0A	00	IP	Indirect Pointer Variable - $4000
	0B	40		
0C	0C	28	LSNADR	Listen Address : Device # + $20
0D	0D	48	TLKADR	Talker Address : Device # + $40
0E	0E	00	LSNACT	Active Listener Flag
0F	0F	00	TLKACT	Active Talker Flag
10	10	00	ADRSED	Addressed Flag
11	11	00	PRGTRK	Last Program Accessed
12	12	01	DRVNUM	Current Drive Number
13	13	00	TRACK	Current Track
14	14	00	SECTOR	Current Sector
15	15	06	LINDX	Logical Index
16	16	0F	SA	Current Secondary Address
17	17	6F	ORGSA	Original Secondary Address
18	18	3F	DATA	Temporary Data Byte
19	19	00	R0	Temp Work Area
1A	1A	00	R1	Temp Work Area
1B	1B	00	R2	Temp Work Area
1C	1C	00	R3	Temp Work Area
1D	1D	00	R4	Temp Work Area
1E-21	1E	00	RESULT	Result of Multiply/Divide Rtns.
	1F	00		
	20	00		
	21	00		
22-26	22	00	ACCUM	Remainder of Multiply/Divide Rtns.
	23	28		
	24	00		
	25	00		
	26	00		
27-28	27	05	DIRBUF	Pointer To Directory Buffer - $4305
	28	43		
29-48	29	00	BUFTAB	Buffer Byte Ptrs. 16 entries, 2 bytes each, point to current byte in corresponding buf.
	29	00		Buffer Byte Ptrs. Buffer #0 Low
	2A	11		High
	2B	00		Buffer #1 Low
	2C	12		High
	2D	00		: Buffer #2 Low
	2E	13		High
	2F	00		: Buffer #3 Low
	30	20		High
	31	00		: Buffer #4 Low
	32	21		High
	33	00		: Buffer #5 Low
	34	22		High
	35	00		: Buffer #6 Low
	36	23		High
	37	00		: Buffer #7 Low
	38	30		High
	39	00		: Buffer #8 Low
	3A	31		High
	3B	00		: Buffer #9 Low
	3C	32		High
	3D	00		: Buffer #10 Low
	3E	33		High
	3F	00		: Buffer #11 Low
	40	40		High
	41	00		: BAM Drive 0 Low
	42	41		: BAM Drive 0 High
	43	00		: BAM Drive 1 Low
	44	42		: BAM Drive 1 High
	45	00		: CMD Buffer Low
	46	43		: CMD Buffer High
	47	DC		: Error Output Buffer Low
	48	43		: Error Output Buffer High
49-50	49	FF	BUF0	Inactive Flags For Buffers. next 16 bytes store buffer pairs for double buffering blocks of seq files. bit7 = 1 indicates inactive buffer, direct access channels use only one buffer: 2nd entry is set to $FF indicating no buffer
	4A	09		
	4B	FF		
	4C	FF		
	4D	FF		
	4E	FF		
	4F	0E		
	50	0F		
51-58	51	FF	BUF1	Active Flags For Buffers. second buffer number of pair associated with channel
	52	88		
	53	FF		
	54	FF		
	55	FF		
	56	FF		
	57	FF		
	58	FF		
59	59	0C	NBKL	Number Of Blocks Low
59-60	59	0C	RECL	Low Record # To Find Relative File
	5A	00		
	5B	00		
	5C	00		
	5D	00		
	5E	00		
	5F	00		
	60	00		
61	61	00	NBKH	Number Of Blocks High
61-68	61	00	RECH	High Record # To Find Relative File
	62	00		
	63	00		
	64	00		
	65	00		
	66	00		
	67	00		
	68	00		
69-70	69	00	NR	Next Record Table
	6A	00		
	6B	00		
	6C	00		
	6D	00		
	6E	00		
	6F	00		
	70	00		
71-78	71	00	RS	Relative Record Size Table
	72	00		
	73	00		
	74	00		
	75	00		
	76	00		
	77	00		
	78	00		
79-80	79	FF	SS	Side Sector Table
	7A	FF		
	7B	FF		
	7C	FF		
	7D	FF		
	7E	FF		
	7F	FF		
	80	FF		
81	81	00	F1PTR	File Stream 1 Pointer
82	82	00	RECPTR	1st Byte Wanted From Relative Record
83	83	00	SSNUM	Side Sector # Of Relative Record
84	84	00	SSIND	Index Into Side Sector
85	85	00	RELPTR	Ptr To 1st Byte Wanted In REL File
86-8A	86	00	ENTSEC	Sector Of Directory Entries. 5 entries, 1 byte each, indicating sector of directory entry for corresponding filename in CMDBUF
	87	00		
	88	00		
	89	00		
	8A	00		
8B-8F	8B	00	ENTIND	Index Of Directory Entries. 5 entries, 1 byte each, indicating the index-2 into sector (from ENTSEC)
	8C	00		
	8D	00		
	8E	00		
	8F	00		
90-94	90	00	FILDRV	Default Flag, Drive Number
	91	00		
	92	00		
	93	00		
	94	00		
95-99	95	00	PATTYP	Pattern, Replace, Closed-Flags, Type
	96	00		
	97	00		
	98	00		
	99	00		
9A-A1	9A	00	FILTYP	Channel File Type. 8 entries, 1 byte each, contains file type times 2 plus drive num. bit7 = 1 indicates search both drives SEQ = type 1 PRG = type 2 USR = type 3 REL = type 4 direct access = type 7
	9B	00		
	9C	00		
	9D	00		
	9E	00		
	9F	00		
	A0	00		
	A1	00		
A2-A9	A2	00	CHNRDY	Channel Status. 8 entries, 1 byte each, indicates channel status for ieee talk and listen sequences. bit7 = 1 channel is talker to ieee, bit3 = 0 send eoi on next byte (talker only), bit0 = 1 channel is listener to ieee, other bits are unused
	A3	01		
	A4	00		
	A5	00		
	A6	00		
	A7	00		
	A8	01		
	A9	88		
AA	AA	20	EOIFLG	Temporary EOI
AB	AB	0A	JOBNUM	Current Job Number
AC-BE	AC	FF	LINTAB	Logical Index Table, contains corresponding secondary address associated with channel number. $FF indicates no active channel. bits 7 and 6 indicate channel direction: 0 0 = read channel 1 0 = write channel 0 1 = read/write channel 1 1 = no channel
	AD	FF		
	AE	FF		
	AF	FF		
	B0	FF		
	B1	FF		
	B2	FF		
	B3	FF		
	B4	FF		
	B5	FF		
	B6	FF		
	B7	FF		
	B8	FF		
	B9	FF		
	BA	FF		
	BB	8F		CMDBUF (write channel)
	BC	0F		Error Channel (read channel)
	BD	FF		
	BE	FF		
BF-C6	BF	82	CHNDAT	Channel Data Byte. contains data byte for output to ieee through GET routines
	C0	00		
	C1	00		
	C2	00		
	C3	00		
	C4	00		
	C5	00		
	C6	30		
C7-CE	C7	FF	LSTCHR	Channel Last Character Pointer. last char pointer in active buf associated with channel. = 0 if not last block in seq file
	C8	00		
	C9	00		
	CA	00		
	CB	00		
	CC	00		
	CD	00		
	CE	E7		
CF	CF	00	TYPE	Active File Type

** The Balance Of Zero Page Is Not Used Directly By DOS **

D0 = 00 D1 = 00 D2 = 00 D3 = 00 D4 = 00 D5 = 00 D6 = 00 D7 = 00
D8 = 00 D9 = 00 DA = 00 DB = 00 DC = 00 DD = 00 DE = 00 DF = 00

E0 = 00 E1 = 00 E2 = 00 E3 = 00 E4 = 00 E5 = 00 E6 = 00 E7 = 00
E8 = 00 E9 = 00 EA = 00 EB = 00 EC = 00 ED = A8 EE = A8 EF = 04

F0 = 80 F1 = 42 F2 = 81 F3 = 53 F4 = 7D F5 = EE, F6 = 7D F7 = EE
F8 = 67 F9 = EF FA = AC FB = EF FC = 34 FD = C4 FE = 78 FF = F2

8050 RAM Memory $0100–

Location	Label	Description
0100-01FF		the stack
0200	IEEEDI	ieee data in
0201	PADD1	ieee data in direction
0202	IEEEDO	ieee data out
0203	PBDD1	ieee data out direction
0204		
0205		
0206		
0207		
0208-027F		unconnected
0280	PAD2	IEEE control port: **
0281	PADD2	**
0282	PBD2	**
0283	PBDD2	**
0284	ATNND	** atn is irq causing ???
0285	ATNPD	**
0286	ATNNE	**
0287	ATNPE	**
0288-0FFF		unconnected
1000	ID	Interrupt Delay (** start of shared memory **)
1001		motor acceleration delay
1002		motor cutoff time
1003-1011	JOBS que	buf #0 Job Codes are:
1004		buf #1 $80 - Read - read t & s specified
1005		buf #2 by header into data buf
1006		buf #3 $90 - Write- write t & s specified
1007		buf #4 by header from data buf
1008		buf #5 $A0 - Verify - compare t & s specified
1009		buf #6 by header with data buf
100A		buf #7 $B0 - Seek - find any header on track
100B		buf #8 specified by hdr. put in data buf
100C		buf #9 $C0 - Bump - track must be set to 1.
100D		buf #10 positions head to track 1
100E		buf #11 $D0 - Jump - jump to user ml code
100F		buf #12 in data buf
1010		buf #13 $E0 - Execute - same as Jump with
1011		buf #14 head in position and drive at speed
1012-1020	TRKS	jobs' track number. used by controller for quick
		reference to track #. must match track in
		corresponding header
1021-10xx	HDRS	job headers for buffers 0-14. 15 entries of 8
		bytes each. controller calculates checksum upon
		execution of job. bits 6 and 7 are used as ID
		extension, currently set at 0 and 0
1021-1022	job header	buf #0 ID1, ID2 Job Error Codes
1023-1024		buf #0 track, sector returned into Job Que
1025-1026		buf #0 checksum, off after Job is executed
1027-1028		buf #0 spare1, spare2 No error : $01
1029-102A	job header	buf #1 ID1, ID2 Can't find header block : $02
102B-102C		buf #1 track, sector No sync character : $03
102D-102E		buf #1 checksum, off Data block not present : $04
102F-1030		buf #1 spare1, spare2 Chksum err in data blk : $05
1031-1032	job header	buf #2 ID1, ID2 not used : $06
1033-1034		buf #2 track, sector Verify error : $07
1035-1036		buf #2 checksum, off Write protect on : $08
1037-1038		buf #2 spare1, spare2 Chksum err in hdr : $09
1039-103A	job header	buf #3 ID1, ID2 Data ran into next hdr : $0A
103B-103C		buf #3 track, sector Disk id mismatch : $0B
103D-103E		buf #3 checksum, off Decoding error : $10
103F-1040		buf #3 spare1, spare2
1041-1048	job header	buf #4 ID1, ID2, trk, sec, chksum, off, 2 spares
1049-1050	job header	buf #5 ID1, ID2, trk, sec, chksum, off, 2 spares
1051-1058	job header	buf #6 ID1, ID2, trk, sec, chksum, off, 2 spares
1059-1060	job header	buf #7 ID1, ID2, trk, sec, chksum, off, 2 spares
1061-1068	job header	buf #8 ID1, ID2, trk, sec, chksum, off, 2 spares
1069-1070	job header	buf #9 ID1, ID2, trk, sec, chksum, off, 2 spares
1071-1078	job header	buf #10 ID1, ID2, trk, sec, chksum, off, 2 spares
1079-1080	job header	buf #11 ID1, ID2, trk, sec, chksum, off, 2 spares
1081-1088	job header	buf #12 ID1, ID2, trk, sec, chksum, off, 2 spares
1089-1090	job header	buf #13 ID1, ID2, trk, sec, chksum, off, 2 spares
1091-1098	job header	buf #14 ID1, ID2, trk, sec, chksum, off, 2 spares
1099-109E	NUMSEC	sectors/track table
109F	VERNUM	dos version number
10A0	ACTJOB	controller's active job
10A1-10A2	PHASE	stepper base phase offset
10A3	STPTRK	number of tracks per step
10A4	NZONES	number of density zones
10A5	SYNDLY	sync delay for pll
10A6-10A7	WPSW	write protect change flag
10A8-10A9	LWPT	last state of write protect switch
10AA	PBI	block identifier
10AB	CFLG2	common flag 2
10AC	NSIDES	number of sides on diskette
10AD-10AF		expand common variables here
10B0	MAXTRK	maximum track number + 1
10B0-10B7	TRKNUM	number of 1st track in each zone but 1st zone
10B8-10BF	OFFSET	recovery track offset for sequential
10C0-10EF		unused ram
10F0-10F1	VNMI	indirect for nmi vector
10F2	NMIFLG	nmi in progress flag
10F3	AUTOFG	auto drive initialization flag
10F4	SECINC	sector increment for sequential files
10F5	REVCNT	error recovery count. set at 10 attempts
10F6-10FF		unused ram
1100	BUFS	start of data buffers
1100-11FF		data buffer #0
1200-12FF		data buffer #1
1300-13FF		data buffer #2
1400-1CFF		unconnected
1D00-1FFF	FBUFS	format download area. code from C000 to CFFF is
		moved here by routine at CC93: format a disk
2000-20FF		data buffer # 3
2100-21FF		data buffer # 4
2200-22FF		data buffer # 5
2300-23FF		data buffer # 6
2400-2FFF		unconnected
3000-30FF		data buffer # 7
3100-31FF		data buffer # 8
3200-32FF		data buffer # 9
3300-33FF		data buffer # 10
3400-3FFF		unconnected
4000-40FF		data buffer # 11
4100-41FF	BAM0	bam drive zero
4200-42FF	BAM1	bam drive one
4300-433A	CMDBUF	command buffer
433B	CMDNUM	command number
433C	STRSIZ	string size in command buffer
433D	TEMPSA	temporary secondary address
433E	CMD	temporary job command
433F	LSTSEC	last sector
4340-4341	BUFUSE	buffer allocation
4342-4343	DSKID	current disk id - drive 0
4344-4345		current disk id - drive 1
4346-4347	MDIRTY	bam dirty flag - drive 0, drive 1
4348	ENTFND	directory entry found flag
4349	DIRLST	directory listing flag
434A	CMDWAT	command waiting flag
434B	LINUSE	logical index (lindx) use word
434C	LBUSED	last buffer used
434D	REC	record size
434E	TRKSS	track of side sector
434F	SECSS	sector of side sector
4350-435E	LSTJOB	15 entries, 1 byte each. last job entered in queue.
		used to retry last job and to extract drive # last used.
435F-4366	DSEC	sector of directory entry
4367-436E	DIND	index of directory entry
436F	ERWORD	error word for recovery
4370	PRGDRV	last program drive
4371	PRGSEC	last program sector
4372	WLINDX	write logical index
4373	RLINDX	read logical index
4374	NBTEMP	number of blocks temporary
4376	CMDSIZ	length of command string + 1
4377	CHAR	character under parser
4378	LIMIT	pointer limit in compar
4379	F1CNT	file stream 1 count
437A	F2CNT	file stream 2 count
437B	F2PTR	file stream 2 pointer
437C-4380	FILTBL	table of filename positions in cmdbuf. 5 entries, 1 byte
		each therefore, 5 filenames max in cmd string.
		corresponding entries point at drive number for
		filename. if present. otherwise first char of filename.
		if d# present, pointer is moved up to 1st char
		of filename after d# is set in TRKS and HDRS unused
4382-4386	FILTRK	track of 1st block in file during searches. bit7 = 1
		indicates pattern matching
4387-438B	FILSEC	sector of 1st block in file during searches.
438C	PATFLG	pattern presence flag
438D	IMAGE	file stream image
438E	DRVCNT	number of drive searches
438F	DRVFLG	drive search flag
4390	LSTDRV	last drive without error
4391	FOUND	found flag in directory searches
4392	DIRSEC	directory sector
4393	DELSEC	sector of 1st available entry
4394	DELIND	index of 1st available entry
4395	LSTBUF	= 0 if last block
4396	INDEX	current index in buffer
4397	FILCNT	counter, file entries
4398	TYPFLG	match by type flag
4399	MODE	active file mode (r, w)
439A	JOBRTN	job return flag
439B	EPTR	pointer for recovery
439C	TOFF	total track offset
439D	NDBL	blocks free - low : drive 0
439E		drive 1
439F	NDBH	blocks free - high : drive 0
43A0		drive 1
43A1	NODRV	no drive flag : drive 0
43A2		drive 1
43A3-43B7		unused ram
43B8-43DB	NAMBUF	directory buffer
43DC-43FF	ERRBUF	error message buffer
4400-BFFF		unconnected

8050 Dual Disk ROM Map

Loc.	Label	Description
C000	CODE	controller format code
C3A1	CDIAG	controller power up diagnostics plus initialization
C421	CCHKSM	checksum .byte 0
C422	PARSXQ	parse and execute string in command buffer
C466	ENDCMD	successful command termination
C470	SCREND	from ENDCMD : scratch entry
C496	CMDERR	command level error processing
C49F	SIMPRS	simple parser
C4B3	PRSCLN	parse colon
C4BC	TAGCMD	tag command string : set up command structure, image and
		file stream pointers
C536	PARSE	parse string : looks for special characters returning when
		variable character is found.
C581	CMDSET	initialize command tables, pointers, etc.
C5AA	CMDRST	clear variables, tables
C5DF	ONEDRV	set 1st drive and table pointers
C5ED	ALLDRS	set up all drives from F2CNT
C609	SETDRV	set drive number
C633	SETANY	set drive from any configuration
C65B	TOGDRV	toggle drive number
C664	FS1SET	set pointers to one file stream and check type
C689	TST0V1	test character in accumulator for '0' or '1'
C696	AUTOIT	rsr test subroutines : check if drvnum drive is inited.
		if catalog calls this routine before any header info is
		transferred, this routine works. routine ends in error if any
		error but disk id occurs
C6D9	OPTSCH	optimal search for lookup and find file
C74F	SCHTBL	search table
		.byt 0,$80,$41,1,1,1,1,$81,$81,$81,$81,$42,$42,$42,$42
C75E	LOOKUP	look up all files in stream and fill tables with info
C79A	FFRE	find next file name matching any file in stream and return
		with entry found stuffed into tables
C7C4	FNDFIL	from FFRE : find file continuous
C7E7	COMPAR	compare all filenames in stream table with each valid entry in
		the directory
C898	CMPCHK	check table for unfound files
C8BB	SRCHST	search directory. returns with valid entry with delind = 0
		or returns with 1st deleted entry with delind = 1
C8BB	SRCHST	initiate a search
C929	SEARCH	continue a search
C94F	AUTOI	auto initialization routines when disk placed in drive
C980	TRNAME	transfer filename from command to buffer
C99A	TRCMBF	transfer command buffer to other buffer
C9B8	FNDLMT	find limit of the string in command buffer
C9E0	GETNAM	get file entry from directory
CAC0	BLKNB	blank name buffer
CACB	NEWDIR	new directory in listing
CB18	MSGFRE	calculate and print the number of blocks free
CB29	FREMSG	.byte 'blocks free.'
CB35	SCRTCH	scratch file(s)
CB8F	DELFIL	delete file by links
CBB7	DELDIR	delete directory entry
CBC2	DUPLCT	duplicate disk
CC0B	CPYD1	copy blocks from one drive to other
CC26	CPYTRK	copy one track
CC4F	READS	read temp + 2 blocks in
CC73	WRITES	write temp + 2 buffers out
CC93	FORMAT	transfer format code to buffer 0 and start controller formatting
CCCD	DSKCPY	checks for type and parses special case
CCEF	DX0000	from DSKCPY : normal parse
CD21	PRSEQ	from DSKCPY : parse seq file
CD48	CPYDTD	copy disk to disk routines
CDDA	TRFNME	transfer name from directory buffer to command buffer
CDEA	PUPS1	set up variables sub-routine
CE07	COPY	copy file(s) to one drive
CE59	CY	from COPY : check files for existance
CE9D	OPIRFL	open & set up read file
CED7	GIBYTE	get in a byte
CEF5	RENAME	rename file name in directory
CF39	CHKIN	from CHKIO
CF53	CHKIO	check i/o file for existance – entrance point
CF64	MEM	memory access commands
CF89	MEMEX	(m–e) memory execute
CF8C	MEMRD	(m–r) memory read
CFB6	MEMERR	memory command error
CFBB	MEMWRT	(m–w) memory write
CFC7	USER	user access commands
CFCE	USRINT	'u0' resets usrjmp vector to point to $ffea
CFD7	US10	execute code by the table. use following rtn to determine
		action
CFDD	USREXC	determine user action to execute and set up accordingly
CFEF	OPNBLK	open direct access buffer from open channel #
D079	BLOCK	block commands
D084	BLK10	bad block command error
D089	BLK30	syntax error
D08E	BLK40	find command
D0A0	BLK60	parse & execute block command
D0BB	BCTAB	block command table .byt 'afrwep'
D0C1	BCJMP	block commands jump table (as follows)
		BLKALC (b–a) : $D15C
		BLKFRE (b–f) : $D153
		BLKRD (b–r) : $D1AF
		BLKWT (b–w) : $D1CC
		BLKEXC (b–e) : $D1FE
		BLKPTR (b–p) : $D218
D0CD	BLKPAR	parse block parameters
D0FF	ASCHEX	convert ascii to hex
D150	DECTAB	decimal table .byt 1,10,100
D153	BLKFRE	(b–f) block–free
D15C	BLKALC	(b–a) block–allocate
D18F	BLKRD2	b–r subroutine
D195	GETSIM	b–r subroutine
D19B	BLKRD3	b–r subroutine
D1AF	BLKRD	(b–r) block–read
D1B8	UBLKRD	user direct read
D1CC	BLKWT	(b–w) block–write
D1F2	UBLKWT	user direct write
D1FE	BLKEXC	(b–e) block–execute
D218	BLKPTR	(b–p) block–pointer
D22D	BUFTST	test for allocated buffer related to secondary address
D24D	BKOTST	test block operation parameters
D250	BLKTST	test for legal block and set up drive, track, and sector
D269	FNDREL	find relative file
D287	MULPLY	multiply : result = rec.# x rec. size + rec. position
D2C9	DIV254	divide : result = quotient, remainder = accum + 1
D2C9	DIV254	divide by 254
D2CC	DIV120	divide by 120
D2D2	DIV100	division routine
D334	ZERRES	zero result
D33D	ACCX4	multiply accum x 4
D340	ACCX2	multiply accum x 2
D348	ADDRES	add accum to result
D355	DBLBUF	toggle active buffer # in bufnum
D37C	PUT	main routine to write to channel
D3B6	PUTBYT	put accum into active buffer of lindx
D3CA	INTDRV	initialize drives command
D3E4	ITRIAL	called for by INITDR
D3F5	INITDR	initialize drive (DRVNUM)
D42A	NF05	calculate free blocks
D45B	STRDBL	start double buffering : use track,sector as starting block
D47F	RDBUF	start a read job on track,sector
D483	WRTBUF	start a write job on track,sector
D4A7	FNDRCH	find read channel

D4C2	FNDWCH	find write channel
D4DF	TYPFIL	get file type
D4E9	GETPRE	entered by getbyt
D4F1	GETBYT	read byte from active buffer and set flag if last data byte
D510	RDBYT	read a character from file and read next block if needed
D557	WRTBYT	write a character and write buffer out to disk if its full
D580	INCPNT	increment pointer of active buffer
D58D	SETDRN	set DRVNUM to drive indicated by LSTJOB of active buffer
D599	GETWCH	sets up buffer number and allocates lindx
D599	GETWCH	entrance for write
D59C	GETRCH	entrance for read
D5E0	FRECHN	free channel associated with secondary address. free read and
		write channels but not channel 15
D600	RELINX	release the lindx
D611	RELBUF	release the buffers
D645	GETBUF	get a free buffer number
D67C	FREBUF	free buffer
D690	CLRCHN	clear channel
D69C	CLDCHN	channel cleared
D6C1	FNDLNX	find a free lindx to use, mark as used in LINUSE
D6DD	GBYTE	get the next character from a channel
D71F	RNDGET	direct file get
D741	SEQGET	sequential file character get
D754	GETERC	get error channel
D78C	NXTBUF	read next buffer of a file
D79F	DRTRD	direct block read
D7A3	DRTWRT	direct block write
D7A5	DRT	actual read/write routine
D7B4	OPNIRD	open internal read channel (secondary address = 16)
D7C4	OPNIWR	open internal write channel (secondary address = 16)
D7CB	NXDRBK	allocate next dir block on track 39 and mark as used in bam
D81B	FREICH	free the internal channel (secondary address = 16)
D829	GETPNT	read the active buffer pointer
D837	DRDBYT	direct read byte
D847	BUFIND	index table of high byte addresses of buffers
		.byte $11, $12, $13
		.byte $20, $21, $22, $23
		.byte $30, $31, $32, $33
		.byte $40, $41, $42, $43
D856	SETLJB	set last job : use lastjob for drive number
D85E	SETJOB	set job up and check track and sector
D89E	TSERR	illegal track or sector
D8B7	TSCHK	track/sector check
D8CA	VNERR	write to wrong version error
D8DF	DOREAD	do job in accum, set up error count and LSTJOB, return when
		job done ok, jmp to error if error on return
D8DF	DOREAD	read entrance point
D8E3	DOWRIT	write entrance point
D8E5	DOJOB	actual do job rtn
D8F2	WATJOB	wait until job(.x) is done then return
D8FF	TSTJOB	test if job(.y) is done yet, if not done return, if ok then return
		else redo it
D913	OK	c = 0 if ok, return
D915	NOTYET	c = 1, not done yet
D981	QUIT	quit routine
D98B	QUIT2	error encountered
D9C6	HEDOFF	set drive head offset
D9E3	MOVHED	move drive head
D9F6	DOREC	do last job recovery
DA1C	SETHDR	set header of active buffer of the current lindx to track, sector.
DA3E	ADDFIL	id add file to directory
E000	ECHKSM	checksum .byte 0 for $E–$F ROM
E001	OPEN	open channel from ieee. parses the input string that is sent as
		an open data channel, load, and save. channels are allocated
		and the dir is searched for filename contained in the string.
E01C	OP02	from OPEN : load last program
E03D	OP021	from OPEN : load directory
E049	OP04	from OPEN : open directory as sequential file
E05F	OP041	from OPEN : open '#' direct access file
E066	OP0415	from OPEN : program file type
E081	OP05	from OPEN : syntax error
E120	OP81	from OPEN : check for replace ('@')
E12F	OP815	from OPEN : bad filename error
E134	OP82	from OPEN : save/write with replace ('@')
E17E	OP90	from OPEN : open read & load
E183	OP95	from OPEN : file not found error
E1A2	OP115	from OPEN : type mismatch error
E1DD	OPREAD	from OPEN : open a read file
E220	OPWRIT	from OPEN : open a write file
E22C	OPFIN	from OPEN : open finished
E246	CKTM	check mode or file type
E24E	CKM1	from CKTM : check mode
E25B	CKT1	from CKTM : check file type
E266	APPEND	append file
E290	LOADIR	load directory
E30D	CLOSE	close the file associated with secondary address
E31C	CLS10	from CLOSE : close directory file
E32C	CLSALL	from CLOSE : close all files
E33A	CLSCHN	from CLOSE : locate and close specific file type
E363	CLSREL	from CLOSE : close relative file
E399	CLSWRT	from CLOSE : close a write channel
E3DC	CLSDIR	directory close on open write file
E47D	OPNRCH	open read channel with 2 buffers
E4EA	INTPNT	initialize variables for open channel
E51C	OPNWCH	open a write channel with 2 buffers
E5CE	PUTSS	put byte into side sector
E5D6	SCFLG	set/clear flags
E5D8	SETFLG	set flag
E5DE	CLRFLG	clear flag
E5E7	TSTFLG	test flag
E5EC	TSTWRT	test write
E5F8	TSTCHN	test for active files from lindx table
E631	SCRUB	write out buffer if dirty
E63D	SETLNK	put track, sector into buffer
E64C	GETLNK	get link from buffer into track and sector
E659	NULLNK	set track link = 0 and sector link = last non-zero character
E66B	SET00	set up pointer to buffer
E67B	CURBLK	read track and sector from header
E692	WRTAB	set up for write in job que, branch to SJ10
E699	RDAB	set up for read in job que, branch to SJ10
E6A0	WRTOUT	set up for write in job que, branch to SJ20
E6A7	RDIN	set up for read in job que, branch to SJ20
E6AE	WRTSS	set up for write in job que, branch to RDS5
E6B5	RDSS	set up for read in job que
E6C1	SJ10	accessed by WRTAB + RDAB
E6CD	SJ20	accessed by WRTOUT + RDIN
E6D7	RDLNK	set track/sector from link in buffer
E6E7	B0TOB0	transfer bytes from one buffer to other
E703	CLRBUF	clear buffer given
E714	SSSET	set side sector pointer to zero
E71E	SSDIR	set DIRBUF with current side sector pointer
E72B	SETSSP	set DIRBUF & BUFTAB with current side sector pointer
E73A	SSPOS	position side sector and BUFTAB to ssnum ssind
E75D	IBRD	indirect block–read
E763	IBWT	indirect block–write
E767	IBOP	code for above routines
E787	GSSPNT	get side sector pointer
E78E	SCAL1	calculate # side sector blocks required
E793	SSCALC	from SCAL1
E79E	ADDT12	add # side sectors needed x 120
E7A8	SSTEST	test ssnum & ssind for residence & range.
E7D5	GETACT	get active buffer number
E7E0	GAFLGS	get active buffer number, set lbused & flags
E7F9	NXTREC	mark end of record then move on to next record
E865	NRBUF	read track, sector link into buffer
E8A5	RELPUT	write relative data into buffer
E8D4	WRTREL	write relative record
E91C	CLREC	put zeros into balance of relative record
E92E	SDIRTY	set dirty flags
E93E	CDIRTY	clear dirty flags
E949	RDREL	read relative file
E996	SETLST	set last character in record
E9D8	FNDLST	find last character in record
E9F1	SSEND	position side sector and BUFTAB to end of last record
EA28	BREAK	illegal system track or sector error encountered
EA2D	RECORD	position relative pointers to given record number or to last
		record if out of range
EA9B	POSITN	position relative data block into active buffer and next block
		into inactive buffer
EAC2	POSBUF	position proper data blocks into buffers
EB00	BHERE	check if required block is in buffer
EB12	NULBUF	set null records in active buffer for extension
EB34	ADDNR	add next record to record size and leave in accum. if c = 1
		then buffer boundary has been crossed
EB4C	ADDREL	add blocks to relative file
EC7B	NEWSS	generate new side sector and fix old side sectors to reflect it
ED29	ERRTAB	error message table
EE37	ETEND	end of error table
EE37	ERMOVE	move error message from ERRTAB to ERRBUF
EE9B	EADV1	error advance and check
EEB3	ERROR	controller error entry (.a = error #, .x = job #)
EEEA	CMDER2	command error
EF29	TLKERR	talker error recovery
EF36	LSNERR	listen error recovery
EF50	HEXDEC	convert hex to bcd
EF60	BCDDEC	convert bcd to decimal
EF71	OKERR	transfer error message to error buffer
EFFB	UTLODR	Utility Loader: used to load user programs or system utilities
		from disk and execute them.
		format: print#15, "&0:filename"
		where file type of filename is 'usr'
		hardware required: connect data and clock line to ground.
		(2–4–5 on connector)
		on entry: only requirement is that the filename of the file to
		be loaded be the first specified name in the command
		buffer (cmdbuf); registers: ignored
		on exit: if the file existed on disk and could be found, and no
		checksum errors were encountered while loading, it is now
		loaded into memory, ready to execute; registers: all destroyed
		execution of the program is started at the first byte loaded
		cmdbuf contains the parameter string for the freshly loaded
		utility or user program
F030	UTLD10	file record fetch loop
F05A	UTLD30	byte storage loop
F091	GTABYT	fetches a byte from the file open on the internal channel.
		checks if this was the last byte in the file. error if it was
F0A3	ADDSUM	adds up checksum into location r1. algorithm:
		newsum = oldsum – newbyte + carry
F0AB	PEZRO	error display routine. blinks the error # + 1 in all three leds
F0D5	DSKINT	intitialize disk for PU10
F0F9	PU10	power up diagnostic
F0F9	PU10	fill zero page accending pattern
F0FF	PU20	then test zero page
F13A	RM10	test two 64k–bit roms ; enter x = start page, exit if ok
F15B	CR20	test all common ram
F1A4	CTEST1	controller test and initialization
F1BE	PERR2	error
F1C1	DIAGOK	diagnostics ok so far
F1D9	INTTAB	initialize buffer pointer table
F252	SETSEC	set up sector/track table depending on the controller used
F25D	SETERR	set up power on error message 'cbm dos v2.5'
F268	IDLE	idle loop: does housekeeping while waiting for job
F2E9	ATNIRQ	atn irq process : irq on atn, listen to pet, clear stack
F3B1	LISTEN	set listen routine : main routine
F43D	LSTRTN	listen routine
F441	TALK	set talk routine : main routine
F47D	TLKRTN	talk routine
F491	STDIR	directory loading function, get the buffer and get it started
F549	MOVBUF	transfer filename to listing buffer
F557	GETDIR	get character from directory loading
F574	VERDIR	validate files with bam, create new bam according to contents
		of files entered in directory
F5D8	VMKBAM	mark bam with file sectors
F600	VUSED	mark track, sector, (BMPNT) as used
F654	USDERR	no block error
F659	VBMASK	bit mask .byte 1,2,4,8,16,32,64,128
F661	VSETB	set bam
F66C	WTMAPS	write bam maps
F6A5	VBMBUF	bam buffer .byte 0,1,2,3
F6A9	NEW	new (format) a diskette
F77B	NEWMAP	build a new map on diskette
F784	NEWMPV	set new bam, called by VERDIR
F805	BAMOUT	set links, version number and write it
F832	MAPOUT	write out the bit map to the drive in LSTJOB (active)
F840	SCRBAM	verify the bam block count matches the bits
F868	NUMFRE	calculate the number of free blocks on drive number
F877	FRETS	mark a track, sector as free in bam
F8A3	DTYBAM	set dirty flag
F8AB	USEDTS	mark track, sector, (BMPNT) as used
F8E8	FREUSE	calculates index into bam for FRETS and USEDTS
F902	BMASK	bit mask table .byte 1,2,4,8,16,32,64,128
F90A	SETMAP	sets up BMPNT,y to bam for track and drive number
F958	JOB2X	set .x = jobnum * 8
F95F	SETBJ	set jobnum = drvnum + bamjob
F967	RDBAM	read 1st bam in
F97C	RDNBAM	read next bam in
F992	MBAM	.y = bamsiz * (track - bmpnt - >bam.lotrk) + mapoff
F9BC	CLRBAM	clear the bam area
F9C5	RDDIR	read directory
F9DC	SETLDS	turn on activity led specified by drvnum
F9F2	ERROFF	turn off error led
F9FB	NXTTS	returns next available track and sector given current t and s
FA2F	NXTERR	from NXTTS : disk full error
FA48	FNDNXT	find the next optimum sector
FA7F	INTTS	returns optimum initial track,sector
FAB4	FNDSEC	find sector
FAC3	DERR	directory error
FAC8	SETBMP	set indirect bam pointer by drvnum
FAD4	GETSEC	set bam and find avaialable sector starting at sector
FAFA	AVCK	bit map validity check
FB2C	MAXSEC	returns number of sectors located on specific track
FB39	KILLP	kill protection
FB46	DIRTRK	directory track number .byt 39
FB47	BAMSIZ	number of bytes/track in bam .byt 5
FB48	MAPOFF	offset of bam in sector .byt 6
FB49	DSKNAM	offset of disk name in bam sector .byt 6
FB4A	BAMTRK	bam track link table .byt 38,38,39
FB4D	BAMSEC	bam sector link table .byt 0,3,1
FB50	CMDTBL	command search table .byt 'vidmbup&crsn'
		(validate, initialize, duplicate, m– , b– , user,
		position, utlodr, copy, rename, scratch, new)
FB5C	CJUMPL	command jump table low bytes
		.byt $74 : VERDIR
		.byt $CA : INTDRV
		.byt $C2 : DUPLCT
		.byt $64 : MEM
		.byt $79 : BLOCK
		.byt $C7 : USER
		.byt $2D : RECORD
		.byt $FB : UTLODR
		.byt $CD : DSKCPY
		.byt $F5 : RENAME
		.byt $35 : SCRTCH
		.byt $A9 : NEW
FB68	CJUMPH	command jump table high bytes
		.byt $F5 : VERDIR
		.byt $D3 : INTDRV
		.byt $CB : DUPLCT
		.byt $CF : MEM
		.byt $D0 : BLOCK
		.byt $CF : USER
		.byt $EA : RECORD
		.byt $EF : UTLODR
		.byt $CC : DSKCPY
		.byt $CE : RENAME
		.byt $CB : SCRTCH
		.byt $F6 : NEW
FB6C	STRUCT	structure images for commands
		.byt %01010001 DSKCPY
		.byt %11011101 RENAME
		.byt %00011100 SCRATCH
		.byt %10011110 NEW
		.byt %00011100 LOAD
FB79	MODLST	mode table .byt 'rwam'
FB7D	TPLST	file type table .byt 'dspul' (DEL, SEQ, PRG, USR, REL)
FB82	TYPLST	1st character in name of file type .byt 'dspur'
FB87	TP1LST	2nd character in name of file type .byt 'eerse'
FB8C	TP2LST	3rd character in name of file type .byt 'lqgrl'
FB91	ER00	error flag variables for bit
FB91	ER00	.byt 0
FB92	ER0	.byt $3F
FB93	ER1	.byt $7F
FB94	ER2	.byt $BF
FB95	ER3	.byt $FF
FB96	IPBM	.byt $41,$42
FB98	DRIVER	numsec (tab1): (4) sectors/track
		.byt 23,25,27,29
		gap1 : header gap, gap2, tail gap (format), vernum: format
		.byt 20,11,fm8050
		actjob, phase(2), stptrk, nzones
		.byt 0,0,0,4,4
		syndly, wpsw(2), lwpt(2), pbi, cflg2, nsides
		.byt 3,1,1,0,0,7,0,1
		unused(3) .byt 0,0,0
		trknum (tab3): zone boundaries track numbers
		.byt 78,65,54,40,0,0,0,0
		offset for recovery
		.byt 1,$FF,$FF,1,2,$FE,$FE,2,0
FBC0	NMI	non maskable interrupt : JMP ($10F0)
FBC3	PATCH	default table for user command
		.byt $80,$50 : 8050
FFEA	UBLOCK	user command set up
		UBLKRD user block read (u1) : $D1B8
		UBLKWT user block write (u2) . $D1F2
		user jmp through (u3) : $1300
		user jmp through (u4) . $1303
		user jmp through (u5) : $1306
		user jmp through (u6) : $1309
		user jmp through (u7) : $130C
		user jmp through (u8) : $130F
FFFA	NMI	kernal nmi : $FBC0
FFFC	DSKINT	kernal disk initialization : $F0D5
FFFE	ATNIRQ	kernal atn irq process : $F2E9

1541 Disk Memory Map

1541 System Constants

Hex Val	Label	Description
$00	LED1	no led on
$00	NOTRDY	i/o not ready
$00	RDMODE	open read mode
$00	VAL	job code for validate
$01	DATIN	data in line
$01	LISNER	serial listener flag
$01	MASK4	bit mask for gcr conversion
$01	RDYLST	ready to listen
$01	SEQTYP	open sequential type
$01	WTMODE	open write mode
$02	APMODE	open append mode
$02	DATOUT	data out
$02	DOSVER	dos version
$02	PRGTYP	open program type
$02	TOLONG	format error : can't find sync mark
$03	MASK7	bit mask for gcr conversion
$03	MDMODE	open modify mode
$03	TOMANY	format error : too many counts
$03	USRTYP	open user type
$04	CLKIN	clock in
$04	CMDCHN	command channel number
$04	GAP2	minimum size of gap after data block
$04	NMODES	number of modes in tables modlst ($FEB6 'rwam')
$04	RELTYP	open relative type
$04	TOBIG	format error : not enough space
$05	BFCNT	available buffer count
$05	ERRCHN	error channel number
$05	MXFILS	maximum number of filenames in string
$05	NTYPES	number of different file types ($FEB8 'dspur')
$05	NUMSYN	gcr byte count for size of sync area
$05	TOSMAL	format error : gap2 too small
$06	BLINDX	bam lindx for floating bams
$06	MXCHNS	maximum number of channels in system
$06	NBCMDS	number of block commands ($CC5D 'afrwep')
$06	NOTFND	format error: file not found
$06	NSSL	number of side-sector links
$06	NUMJOB	number of jobs
$06	RDMAX	sector distance wait
$07	DIRTYP	open direct file type
$07	MASK2	bit mask for gcr conversion
$07	TYPMSK	mask for type bits
$07	VERERR	controller verify error
$08	CLKOUT	clock out
$08	EOISND	not(eoi) to send
$08	EOI	not(eoi) to send
$08	LED0	active led
$09	GAP1	gap after header to clear erase in gcr
$09	WRTMIN	write minimum
$0A	CBPTR	command buffer pointer
$0C	LDCMD	load command image
$0C	MSGLEN	length of 'blocks free' message at $C817
$0C	NCMDS	number of commands ('vidmbup&crsn')
$0C	WRTMAX	write maximum
$0D	CR	carriage return
$0F	CMDSA	command channel secondary address number
$0F	LXINT	power up logical index usage (linuse)
$0F	MASK5	bit mask for gcr conversion
$10	ATNA	atn active
$10	ERRSA	error channel secondary address number
$10	SSIOFF	offset into ss for data block pointers
$11	IRSA	internal read secondary address number
$12	IWSA	internal write secondary address number
$12	MAXSA	maximum secondary address number plus one
$18	DIRLEN	directory length used
$1B	NBSIZ	nambuf text size
$1F	MASK8	bit mask for gcr conversion
$20	OVRFLO	rr print overflow
$29	CMDLEN	length of command buffer
$2C	SKIP2	bit abs
$30	BADSYN	error: general syntax
$31	BADCMD	error: invalid command
$32	LONGLN	error: long line
$33	BADFN	error: invalid filename
$34	NOFILE	error: no file given
$39	NOCFIL	error: command file not found
$3A	TIM	irq rate for 15 ms
$3E	MASK3	bit mask for gcr conversion
$3F	UNLSN	unlisten command
$40	BUMPC	bump command
$40	DYFILE	dirty flag for rr file
$41	FM4040	4040 format version
$42	FM2030	2030 format version
$45	TOPRD	top of read overflow buffer on a read
$45	TOPWRT	top of write overflow buffer in a write
$50	JMPC	jump command
$50	NOREC	error: record not present
$51	RECOVF	error: overflow in record
$52	BIGFIL	error: file too large
$5F	UNTLK	untalk command
$60	EXECD	execute command
$60	FILOPN	error: file open
$61	FILNOP	error: file not open
$62	FLNTFD	error: file not found
$63	FLEXST	error: file exists error
$64	MISTYP	error: file type mismatch
$65	NOBLK	error: no block
$66	BADTS	error: illegal track or sector
$67	SYSTS	error: illegal system track or sector
$70	NOCHNL	error: no channels available
$71	DIRERR	error: directory error
$72	DSKFUL	error: diskette full
$73	CBMV2	'cbm dos v2.6 v170' message number
$74	NODRIV	error: drive not ready
$78	NSSP	number of pointers in side sector
$7D	MASK6	bit mask for gcr conversion
$80	ATN	atn in
$80	EOIOUT	talk with eoi
$80	LRF	last record flag
$80	MASKSX	bit mask for gcr conversion
$80	READ	controller job type: read
$80	TALKER	talker flag
$81	RNDEOI	random with eoi
$88	RDYTLK	talk no eoi
$89	RNDRDY	random chnrdy
$90	WRITE	controller job type: write
$A0	WVERFY	controller job type: verify
$B0	SEEK	controller job type: seek
$C0	BUMP	controller job type: bump
$C0	MASK2X	bit mask for gcr conversion
$D0	JUMPC	controller job type: jump
$E0	EXEC	controller job type: execute
$E0	MASK7X	bit mask for gcr conversion
$F0	MASK4X	bit mask for gcr conversion
$F8	MASK1	bit mask for gcr conversion

1541 RAM Memory Map with Zero Page Contents at Power Up

references to Drive 1 are mostly unused locations

Hex Location		Content	CBM Label	Function
00–05	00	00	JOBS	Job Que: Buffer #0
	01	00		Buffer #1
	02	00		Buffer #2
	03	00		Buffer #3
	04	00		Buffer #4
	05	00		Buffer #5
06–11	06	00	HDRS	Job Headers: Buffer #0 - Low
	07	00		Buffer #0 - High
	08	00		Buffer #1 - Low
	09	00		Buffer #1 - High
	0A	00		Buffer #2 - Low
	0B	00		Buffer #2 - High
	0C	00		Buffer #3 - Low
	0D	00		Buffer #3 - High
	0E	00		Buffer #4 - Low
	0F	00		Buffer #4 - High
	10	00		Buffer #5 - Low
	11	00		Buffer #5 - High
12–15	12	00	DSKID	Master Copy Of Disk ID: Drive 0
	13	00		Drive 0
	14	00		Not Used - Drive 1
	15	00		Not Used - Drive 1
16–1A	16	00	HEADER	Image Of Last Header: ID Byte 1
	17	00		ID Byte 2
	18	00		Track
	19	00		Sector
	1A	00		Checksum
1B	1B	00	ACTJOB	Controllers Active Job
1C–1D	1C	01	WPSW	Write Protect Change Flag: Drive 0
	1D	01		Drive 1
1E–1F	1E	10	LWPT	Last State Of WP Switch: Drive 0
	1F	00		Drive 1
20	20	00	DRVST	Drives Current Status: Drive 0
21	21	00		Speed Timing Flag
22–23	22	00	DRVTRK	Drive Track Number: Drive 0
	23	00		Drive 1
24–2D	24	00	STAB	Storage Table For GCR Conversion
	25	00		
	26	00		
	27	00		
	28	00		
	29	00		
	2A	00		
	2B	00		
	2C	00		
	2D	00		
2E–2F	2E	00	SAVPNT	Temporary Save Pointer Location
	2F	00		
30–31	30	00	BUFPNT	Active Buffer Pointer
	31	00		
32–33	32	00	HDRPNT	Header Pointer: Track
	33	00		Sector
34	34	00	GCRPNT	GCR Pointer
35	35	00	GCRERR	Indicates GCR Decode Error
36	36	00	BYTCNT	Byte Counter For GCR/Binary Conv
37	37	00	BITCNT	Bit Counter
38	38	00	BID	Data Block ID
39	39	00	HBID	Header Block ID
3A	3A	00	CHKSUM	Checksum
3B	3B	00	HINIB	•not used directly
3C	3C	00	BYTE	•not used directly
3D	3D	00	DRIVE	Drive Number
3E	3E	FF	CDRIVE	Current Active Drive Number
3F	3F	00	JOBN	Current Job Number
40	40	00	TRACC	Track - Internal Storage Location
41	41	00	NXTJOB	Next Job
42	42	00	NXTRK	Next Track
43	43	00	SECTR	Sector Per Track For Formatting
44	44	00	WORK	Working Storage Location
45	45	00	JOB	Job Type
46	46	00	CTRACK	•not used directly
47	47	07	DBID	Data Block ID
48	48	00	ACLTIM	Accel Time Delay
49	49	39	SAVSP	Save Stack Pointer
4A	4A	00	STEPS	Steps To Desired Track
4B	4B	00	TMP	Temporary Storage Location
4C	4C	00	CSECT	Current Sector
4D	4D	00	NEXTS	Next Sector
4E	4E	00	NXTBF	Pointer To Next GCR Source Buffer
4F	4F	00	NXTPNT	Ptr To Next Byte Location In Buffer
50	50	00	GCRFLG	GCR/Binary Flag In Active Buffer
51	51	FF	FTNUM	Current Format Track
52–55	52	00	BTAB	Binary Table: GCR/Binary Work Area
	53	00		
	54	00		
	55	00		
56–5D	56	00	GTAB	GCR Table: GCR/Binary Work Area
	57	00		
	58	00		
	59	00		
	5A	00		
	5B	00		
	5C	00		
	5D	00		
5E	5E	04	AS	Number Of Steps To Accel With Head
5F	5F	04	AF	Acceleration Factor
60	60	00	ACLSTP	Steps To Go Before Complete
61	61	00	RSTEPS	Number Of Run Steps
62–63	62	05	NXTST	Pointer To Stepping Rtn - $FA05
	63	FA		
64	64	C8	MINSTP	Minimum Steps Required To Accel
65–66	65	22	VNMI	Indirect For NMI - $EB22
	66	EB		
67	67	00	NMIFLG	NMI In Progress Flag
68	68	00	AUTOFG	Auto Drive Initialization Flag
69	69	0A	SECINC	Sector Increment For Sequential
6A	6A	05	REVCNT	Error Recovery Count
6B–6C	6B	EA	USRJMP	User Jump Table Pointer - $FFEA
	6C	FF		
6D–6E	6D	00	BMPNT	Bit Map Pointer
	6E	00		
6F–74	6F	6F	TEMP: T0	Temporary Work Space
	70	00	T1	
	71	00	T2	
	72	FF	T3	
	73	00	T4	
	74	00		
75–76	75	00	IP	Indirect Pointer Variable
	76	01		
77	77	28	LSNADR	Listen Address: Device # + $20
78	78	48	TLKADR	Talker Address: Device # + $40
79	79	00	LSNACT	Active Listener Flag
7A	7A	00	TLKACT	Active Talker Flag
7B	7B	00	ADRSED	Addressed Flag
7C	7C	00	ATNPND	Attention Pending Flag
7D	7D	00	ATNMOD	In ATN Mode
7E	7E	00	PRGTRK	Last Program Accessed
7F	7F	00	DRVNUM	Current Drive Number
80	80	00	TRACK	Current Track
81	81	00	SECTOR	Current Sector
82	82	04	LINDX	Logical Index
83	83	0F	SA	Current Secondary Address
84	84	6F	ORGSA	Original Secondary Address
85	85	3F	DATA	Temporary Data Byte
86	86	00	R0	Temp Work Area
87	87	00	R1	Temp Work Area
88	88	00	R2	Temp Work Area
89	89	00	R3	Temp Work Area
8A	8A	00	R4	Temp Work Area
8B–8E	8B	00	RESULT	Result Of Multiply/Divide Rtns
	8C	00		
	8D	00		
	8E	00		
8F–93	8F	00	ACCUM	Remainder Of Multiply/Divide Rtns
	90	00		
	91	00		
	92	00		
	93	00		
94–95	94	04	DIRBUF	Pointer To Directory Buffer
	95	02		
96	96	00	ICMD	IEEE Command In Not Used
97	97	06	MYPA	MY PA Flag: Not Used
98	98	00	CONT	Serial Bit Counter
99–A6	99	00	BUFTAB	Buffer Byte Ptrs: Buffer #0 Low
	9A	03		: Buffer #0 High
	9B	00		: Buffer #1 Low
	9C	04		: Buffer #0 High
	9D	00		: Buffer #2 Low
	9E	05		: Buffer #0 High
	9F	00		: Buffer #3 Low
	A0	06		: Buffer #0 High
	A1	00		: Buffer #4 Low
	A2	07		: Buffer #0 High
	A3	00		: CMD Buffer Low
	A4	02		High
	A5	D6		: Error Buff Low
	A6	02		High
A7–AD	A7	FF	BUF0	Inactive Flags For Buffers
	A8	FF		
	A9	FF		
	AA	FF		
	AB	05		
	AC	06		
	AD	FF		
AE–B4	AE	FF	BUF1	Active Flags For Buffers
	AF	FF		
	B0	FF		
	B1	FF		
	B2	FF		
	B3	FF		
	B4	FF		
B5	B5	00	NBKL	Number Of Blocks Low
B5–BA	B5	00	RECL	Low Record # To Find Relative File
	B6	00		
	B7	00		
	B8	00		
	B9	00		
	BA	00		
BB	BB	00	NBKH	Number Of Blocks High
BB–C0	BB	00	RECH	High Record # To Find Relative File
	BC	00		
	BD	00		
	BE	00		
	BF	00		
	C0	00		
C1–C6	C1	00	NR	Next Record Table
	C2	00		
	C3	00		
	C4	00		
	C5	00		
	C6	00		
C7–CC	C7	00	RS	Relative Record Size Table
	C8	00		
	C9	00		
	CA	00		
	CB	00		
	CC	00		
CD–D2	CD	FF	SS	Side Sector Table
	CE	FF		
	CF	FF		
	D0	FF		
	D1	FF		
	D2	FF		
D3	D3	00	FIPTR	File Stream 1 Pointer
D4	D4	00	RECPTR	1st Byte Wanted From Relative Record
D5	D5	00	SSNUM	Side Sector Number Of Relative File
D6	D6	00	SSIND	Index Into Side Sector
D7	D7	00	RELPTR	Ptr To 1st Byte Wanted In Rel File
D8–DC	D8	00	ENTSEC	Sector Of Directory Entries
	D9	00		
	DA	00		
	DB	00		
	DC	00		
DD–E1	DD	00	ENTIND	Index Of Directory Entries
	DE	00		
	DF	00		
	E0	00		
	E1	00		
E2–E6	E2	00	FILDRV	Default Flag, Drive Number
	E3	00		
	E4	00		
	E5	00		
	E6	00		
E7–EB	E7	00	PATTYP	Pattern, Replace, Closed-Flags, Type
	E8	00		
	E9	00		
	EA	00		
	EB	00		
EC–F1	EC	00	FILTYP	Channel File Type
	ED	00		
	EE	00		
	EF	00		
	F0	00		
	F1	00		
F2–F7	F2	60	CHNRDY	Channel Status
	F3	00		
	F4	00		
	F5	00		
	F6	01		
	F7	88		
F8	F8	80	EOIFLG	Temporary EOI
F9	F9	00	JOBNUM	Current Job Number
FA–FE	FA	00	LRUTBL	Least Recently Used Buffer Table
	FB	01		
	FC	02		
	FD	03		
	FE	06		
FF–100	FF	00	NODRV	No Drive Flag: Drive 0

1541 RAM Memory $0100–

Location	Label	Description
0101–0102	DSKVER	disk version from 18.0
0103	ZPEND	- not used
0104–01FF		the stack
0200–0229	CMDBUF	command buffer
022A	CMDNUM	command number
022B–202D	LINTAB	secondary address : logical index table
023E–0243	CHNDAT	channel data byte
0244–0249	LSTCHR	channel last character pointer
024A	TYPE	active file type
024B	STRSIZ	string size in command buffer
024C	TEMPSA	temporary secondary address
024D	CMD	temporary job command
024E	LSTSEC	last sector
024F	BUFUSE	buffer allocation
0251–0252	MDIRTY	bam dirty flag : drives 0 and 1
0253	ENTFND	directory entry found flag
0254	DIRLST	directory listing flag
0255	CMDWAT	command waiting flag
0256	LINUSE	logical index (lindx) use word
0257	LBUSED	last buffer used
0258	REC	record size
0259	TRKSS	track of side sector
025A	SECSS	sector of side sector
025B–025F	LSTJOB	last job
0260–0265	DSEC	sector of directory entry
026C–026B	DIND	index of directory entry
026C	ERWORD	error word for recovery
026D	ERLED	error led mask for flashing
026E	PRGDRV	last program drive
026F	PRGSEC	last program sector
0270	WLINDX	write logical index
0271	RLINDX	read logical index
0272–0273	NBTEMP	number blocks temporary
0274	CMDSIZ	command string size
0275	CHAR	character under parser
0276	LIMIT	pointer limit in compar
0277	F1CNT	file stream 1 count
0278	F2CNT	file stream 2 count
0279	F2PTR	file stream 2 pointer
027A–027F	FILTBL	filename pointer
0280–0284	FILTRK	1st link/track
0285–0289	FILSEC	1st link/sector
028A	PATFLG	pattern presence flag
028B	IMAGE	file stream image
028C	DRVCNT	number of drive searches
028D	DRVFLG	drive search flag
028E	LSTDRV	last drive without error
028F	FOUND	found flag in directory searches
0290	DIRSEC	directory sector
0291	DELSEC	sector of 1st available entry
0292	DELIND	index of 1st available entry
0293	LSTBUF	= 0 if last block
0294	INDEX	current index in buffer
0295	FILCNT	counter, file entries
0296	TYPFLG	match by type flag
0297	MODE	active mode (r, w)
0298	JOBRTN	job return flag
0299	EPTR	pointer for recovery
029A	TOFF	total track offset
029B–029C	UBAM	last bam update pointer
029D–029E	TBAM	track number of bam image
02A1–02B0	BAM	bam images
02B1–02D4	NAMBUF	directory buffer
02D5–02F8	ERRBUF	error message buffer
02F9	WBAM	don't-write-bam flag
02FA–02FB	NDBL	blocks free low byte: drive 0 and 1
02FC–02FD	NDBH	blocks free high byte: drive 0 and 1
02FE–02FF	PHASE	phase offset
0300	BUFS	start of data buffers
0300	FBUFS	format download image
0300–03FF	BUFF0	buffer #0
0400–04FF	BUFF1	buffer #1
0500–05FF	BUFF2	buffer #2
0600–06FF	BUFF3	buffer #3
0620	CNT	error counter: decrements from 10
0620	FMTVAR	format variable
0621	NUM	number between sync and non-sync
0623	TRYS	number of tries in verify
0624–0625	TRAL	
0626	DTRCK	distance to track
0627	REMDR	remainder of size
0628	SECT	sector number counter
1800	PB	data port b
1801	PA1	data port a - unused
1802	DDRB1	data direction register port b
1803	DDRA1	data direction register port a
1804	T1LC1	timer 1 low counter
1805	T1HC1	timer 1 high counter
1805	TIMER1	timer one counter
1806	T1LL1	timer 1 low latch
1807	T1HL1	timer 1 high latch
1808	T2LC1	timer 2 low counter
1809	T2HC1	timer 2 high counter
180A	SR1	shift register
180B	ACR1	auxillary control register
180C	PCR1	peripheral control register
180D	IFR1	interrupt flag register
180E	IER1	interrupt enable register
1C00	DSKCNT	disk controller i/o control line
		bit 0: step head in
		bit 1: step head out
		bit 2: motor on
		bit 3: act led
		bit 4: write protect sense
		bit 5: density select 0
		bit 6: density select 1
		bit 7: sync detect
1C01	DATA2	data port a
1C02	DDRB2	data direction for port b
1C02	LEDOUT	ddrb of $1c00 for output led
1C03	DDRA2	data direction for port a
1C04	T1LC2	timer 1 low counter
1C05	T1HC2	timer 1 high counter
1C06	T1LL2	timer 1 low latch
1C07	T1HL2	timer 1 high latch
1C08	T2LL2	timer 2 low latch
1C09	T2LH2	timer 2 high latch
1C0A	SR2	shift register
1C0B	ACR2	auxillary control register
1C0C	PCR2	peripheral control register
1C0D	IFR2	interrupt flag register
1C0E	IER2	interrupt enable register

1541 Disk ROM Map

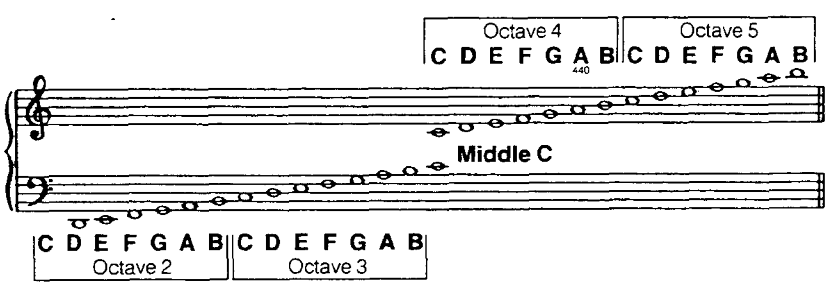
Loc.	Label	Description
C000	ROM	start of rom
C001	FREEC0	(-C0FF) controller code patch space
C100	SETLDS	turn on activity led specified by drive number
C123	ERROFF	turn off error led
C12C	ERRON	turn on error led
C146	PARSXQ	parse and execute string in command buffer
C194	ENDCMD	successful command termination
C1BD	CLRCB	clear command buffer
C1C8	CMDERR	command level error processing
C1D1	SIMPRS	simple parser
C1E5	PRSCLN	find position of colon
C1EE	TAGCMD	tag command string : set up command structure, image & file stream pointers
C268	PARSE	parse string : looks for special characters returning when variable character is found
C2B3	CMDSET	initialize command tables, pointers, etc.
C2DC	CMDRST	clear variables, tables
C312	ONEDRV	set up 1ST drive and table pointers
C320	ALLDRS	set up all drives from f2cnt
C33C	SETDRV	set drive number
C368	SETANY	set drive from any configuration
C38F	TOGDRV	toggle drive number
C398	FS1SET	set pointers to one file stream and check type
C3BD	TST0V1	test character in accumulator for '0' or '1'
C3CA	OPTSCH	optimal search for lookup and fndfil
C440	SCHTBL	search table
		; .byt 0,$80,$41
		; .byt 1,1,1,1
		; .byt $81,$81,$81,$81
		; .byt $42,$42,$42,$42
C44F	LOOKUP	look up files in stream and fill tables with information
C48B	FFRE	find next file name matching any file in stream and return with entry found stuffed into tables
C4B5	FNDFIL	— " —
C4D8	COMPAR	compare all file names in stream table with each valid entry in the directory
C589	CMPCHK	check table for unfound files
C5AC	SRCHST	search directory : returns with valid entry with delind = 0 or returns with 1ST deleted entry with delind = 1
C5AC	SRCHST	initiate search
C617	SEARCH	continue search
C63D	AUTOI	check drive for active diskette, initialize if needed, return nodrv status
C66E	TRNAME	transfer filename from command to buffer A: string size X: starting index in cmdbuf Y: buffer number
C688	TRCMBF	transfer command buffer to other buffer : uses current buffer pointer limit: ending index + 1 in command buffer X: starting index in command buffer Y: buffer number
C6A6	FNDLMT	find the limit of the string in cmdbuf pointed to by x
C6CE	GETNAM	get file entry from directory
C7AC	BLKNB	blank name buffer
C7B7	NEWDIR	new directory in listing
C806	MSGFRE	display 'blocks free' message
C817	FREMSG	.byt 'blocks free'
C823	SCRTCH	scratch file(s)
C87D	DELFIL	delete file by links
C8B6	DELDIR	delete directory entry
C8C1	DUPLCT	duplicate diskette
C8C6	FORMAT	transfer format control to buf#0, start controller formatting
C8F0	DSKCPY	check for type and parses special case
C932	PUPS1	set up subroutine
C952	COPY	copy file(s) to one file
C9A7	CY	check if file exists
C9FA	OPIRFL	open internal read file
CA35	GIBYTE	get a byte (internal set up)
CA39	GCBYTE	get a byte
CA53	CYEXT	copy relative records
CA88	RENAME	rename file name in directory
CACC	CHKIN	check i/o file for existance (chkio entrance)
CAF8	MEM	memory access commands
CB1D	MEMEX	memory-execute (m-e)
CB20	MEMRD	memory-read (m-r)
CB4B	MEMERR	bad command error
CB50	MEMWRT	memory-write (m-w)
CB5C	USER	user commands
CB63	USRINT	user jump initialize
CB6C	US10	user code entrance for execution
CB72	USREXC	user code execution from table
CB84	OPNBLK	open direct access buffer from open buffer #
CC1B	BLOCK	block commands
CC26	BLK10	bad command error
CC2B	BLK30	bad syntax error
CC30	BLK40	find command
CC42	BLK60	execute command
CC5D	BCTAB	.byt 'afrwep'
CC63	BCJMP	block jump table
		$CD03 BLKALC block-allocate (b-a)
		$CCF5 BLKFRE block-free (b-f)
		$CD56 BLKRD block-read (b-r)
		$CD73 BLKWT block-write (b-w)
		$CDA3 BLKEXC block-execute (b-e)
		$CDBD BLKPTR block-pointer (b-p)
CC6F	BLKPAR	parse block parameters
CCA1	ASCHEX	convert ascii to hex and store conversion in tables
CCF2	DECTAB	decimal table .byt 1,10,100
CCF5	BLKFRE	block-free (b-f)
CD03	BLKALC	block-allocate (b-a)
CD36	BLKRD2	(b-r) subroutine
CD3C	GETSIM	(b-r) subroutine
CD42	BLKRD3	(b-r) subroutine
CD56	BLKRD	block-read (b-r)
CD5F	UBLKRD	user direct read
CD73	BLKWT	block-write (b-w)
CD97	UBLKWT	user direct write
CDA3	BLKEXC	block-execute (b-e)
CDBD	BLKPTR	block-pointer (b-p)
CDD2	BUFTST	test for allocated buffer related to secondary address
CDF2	BKOTST	test block operation parameters
CDF5	BLKTST	test for legal block and set up drive, track, sector
CE0E	FNDREL	find relative file ; inputs ; RECL 1 byte = low record number ; RECH 1 byte = high record number ; RS 1 byte = record size ; RECPTR 1 byte = first byte wanted from record ; outputs ; SSNUM 1 byte = side sector number ; SSIND 1 byte = index into side sector ; RELPTR 1 byte = pointer to first byte wanted
CE2C	MULPLY	multiply : result = rec.number x rec.size + rec.pointer
CE6E	DIV254	divide : result = quotient, remainder = accumulator + 1
CE6E	DIV254	divide by 254
CE71	DIV120	divide by 120
CE77	DIV100	— " —
CE87	DIV150	— " —
CE89	DIV200	divide by 256
CEA3	DIV300	divide
CEB0	DIV400	— " —
CEBF	DIV500	— " —
CED6	DIV600	— " —
CED8	DIV700	— " —
CED9	ZERRES	zero result
CEE2	ACCX4	multiply accumulator X 4
CEE5	ACCX2	multiply accumulator X 2
CEED	ADDRES	add accumulator to result: result = result + accum + 1,2,3
CEFA	LRUINT	initialize the lru table
CEFC	LRUILP	least recently used table update
CF1E	DBLBUF	double buffer routine to switch the active and inactive buffers
CF76	DBL30	error - no buffers
CF7B	DBSET	double buffer set
CF8C	TGLBUF	toggle the inactive and active buffers
CF9B	PIBYTE	
CFAF	PBYTE	
CFB7	PUT	main routine to write to channel
CFF1	PUTBYT	put accumulator into active buffer of logical index
D005	INTDRV	initialize drives (command)
D00E	ITRIAL	initialize drive (drvnum)
D075	NFCALC	count number of free blocks
D09B	STRRD	start double buffering, use track, sector as starting block
D0C3	RDBUF	start a read job on track, sector entry point
D0C7	WRTBUF	start a write job on track, sector entry point
D0C9	STRTIT	actual job routine
D0EB	FNDRCH	find read channel
D107	FNDWCH	find write channel
D125	TYPFIL	get file type
D12F	GETPRE	set up x,y from active buffer number
D137	GETBYT	read a byte from active buffer and set flag if last data byte
D156	RDBYT	read a character from file and read next block of file if needed
D19D	WRTBUF	write a character to channel and write buffer to disk if full
D1C6	INCPNT	increment pointer of active buffer by accumulator
D1D3	SETDRN	set drvnum to drive idicated by lstjob of active buffer
D1DF	GETWCH	set up buffer number and allocates logical index
D1DF	GETWCH	write entry point
D1E2	GETRCH	read entry point
D1E3	GETR2	main routine for above
D227	FRECHN	free channel associated with secondary address, free read & write channels, don't free channel #15
D249	RELINX	release the logical index
D25A	RELBUF	given secondary address, free its read channel, release buffers
D28E	GETBUF	get a free buffer number
D2BA	FNDBUF	find a free buffer number and set bit in bufuse
D2DA	FREIAC	free inactive buffer
D307	CLRCHN	clear channel
D313	CLDCHN	cleared channel
D339	STLBUF	steal a buffer: search the channels in order of least recently used and steal the first inactive buffer found
D37F	FNDLNX	find a free logical index to use, mark as used in linuse
D39B	GBYTE	get next character from a channel
D3AA	GET	— " —
D3DE	RNDGET	get character from direct file
D400	SEQGET	get character from sequential file
D409	GET6	is a load
D414	GETERC	get error channel
D44D	NXTBUF	read next buffer of a file, follow links in first two bytes, end of file if 1st byte = 0, 2nd byte length
D460	DRTRD	direct block read entry point
D464	DRTWRT	direct block write entry point
D466	DRT	routine for block read/write
D475	OPNIRD	open internal read channel (sa = 16)
D486	OPNIWR	open internal write channel (sa = 16)
D48D	NXDRBK	allocate next directory block on 18 and mark as used in bam
D4C8	SETPNT	set new pointer
D4DA	FREICH	free internal channel (sa = 16)
D4E8	GETPNT	read the active buffer pointer
D4EB	SETDIR	— " —
D4F6	DRDBYT	direct read byte accumulator = byte number to read
D506	SETLJB	set lastjob
D50E	SETJOB	set job up and check track & sector
D54A	TSERR	illegal track & sector
D55F	TSCHK	track/sector check
D572	VNERR	write to wrong version error
D57A	SJB1	not write, restore
D586	DOREAD	do job in accumulator, set up error count and istjob, return when job done ok, jump to error if error returns
D586	DOREAD	read entry point
D58A	DOWRIT	write entry point
D58C	DOJOB	do job routine
D599	WATJOB	wait until job(.x) is done, return after done
D5A6	TSTJOB	test if job(.x) is done, if not then return, if ok then return else redo it
D5C2	OK	c = 0 if ok, return
D5C4	NOTYET	c = 1, not done yet
D635	QUIT	quit routine
D63F	QUIT2	error encountered
D644	REC7	from tstjob
D676	HEADOFF	set drive head offset

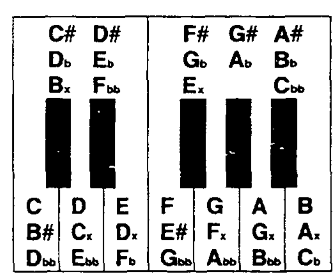
D693	MOVHED	move drive head
D6A6	DOREC	do last job recovery
D6D0	SETHDR	set header of active buffer of the current lindx to track, sector, id
D6E4	ADDFIL	add file to directory
D7B4	OPEN	open channel from ieee. parses the input string that is sent as
		an open data channel. load. or save. channels are allocated.
		the dir is searched for the filename contained in the string
D7CF	OP02	(↑): load last program
D7EB	ENDRD	— " —
D7F3	OP021	(↑): load directory
D7FF	OP04	(↑): open directory as sequential file
D815	OP041	(↑): open '*' direct access file
D81C	OP0415	(↑): program type file
D837	OP05	(↑): syntax error
D8E1	OP81	(↑): check for replace (@)
D8F0	OP815	(↑): bad filename error
D8F5	OP82	(↑): save/write with replace (@)
D940	OP90	(↑): open read & load
D945	OP95	(↑): file not found error
D965	OP115	(↑): type mismatch error
D9A0	OPREAD	(↑): open a read file
D9E3	OPWRIT	(↑): open a write file
D9EF	OPFIN	(↑): open finished
DA09	CKTM	check mode or file type
DA1C	CKM2	check mode
DA1E	CKT1	check type
DA2A	APPEND	append file
DA55	LOADIR	load directory
DAC0	CLOSE	close the file associated with secondary address
DAD4	CLS10	close directory file
DAE9	CLS15	error – free internal channel
DAEC	CLSALL	close all files
DAFF	CLS25	error – free internal channel
DB02	CLSCHN	locate & close specific file type
DB2C	CLSREL	close relative file
DB62	CLSWRT	close write file
DBA5	CLSDIR	directory close on open write file
DC46	OPNRCH	open read channel with 2 buffers
DCB6	INITP	initialize variables for open channel
DCDA	OPNWCH	open write channel with 2 buffers
DD8D	PUTSS	put byte into side sector
DD95	SCFLG	set/clear flags
DD97	SETFLG	set flag
DD9D	CLRFLG	clear flag
DDA6	TSTFLG	test flag
DDAB	TSTWRT	test write
DDB7	TSTCHN	test for active files from lindx table
DDF1	SCRUB	write out buffer if dirty
DDFD	SETLNK	put track, sector into buffer
DE0C	GETLNK	get link from buffer into track & sector
DE19	NULLNK	set track link = 0 & sector link = last non-zero character
DE2B	SET00	set up pointer to buffer
DE3B	CURBLK	read track, sector from header
DE3E	GETHDR	— " —
DE50		do read and write jobs
DE50	WRTAB	set up for write in job que, branch to sj10
DE57	RDAB	set up for read in job que, branch to sj10
DE5E	WRTOUT	set up for write in job que, branch to sj20
DE65	RDIN	set up for read in job que, branch to sj20
DE6C	WRTSS	set up for write in job que, branch to rds5
DE73	RDSS	set up for read in job que
DE95	RDLNK	set track/sector from link in buffer
DEA5	B0TOB0	transfer bytes from one buffer to other
DEC1	CLRBUF	clear buffer given
DED2	SSSET	set side sector pointer to zero
DEDC	SSDIR	set dirbuf with current side sector pointer
DEE9	SETSSP	set dirbuf & buftab with current side sector pointer
DEF8	SSPOS	position side sector & buftab to ssnum ssind
DF1B	IBRD	indirect block-read
DF21	IBWT	indirect block-write
DF25	IBOP	code for above rtns
DF45	GSSPNT	get side sector pointer
DF4C	SCAL1	calculate side sectors
DF5C	ADDT12	— " —
DF65	ADDRTS	— " —
DF66	SSTEST	test ssnum & ssind for residence & range
DF93	GETACT	get active buffer number
DF9E	GAFLGS	get active buffer number, set lbused & flags
DFB7	GETINA	get inactive channels buffer numbers
DFC2	PUTINA	put inactive buffer
DFD0	NXTREC	go to next relative record
E03C	NRBUF	read into buffer
E07C	RELPUT	write relative data to buffer
E0AB	WRTREL	write relative record
E0F3	CLREC	put zeros into balance of relative record buffer
E105	SDIRTY	set dirty flag
E115	CDIRTY	clear dirty flag
E120	RDREL	read relative record
E16E	SETLST	set last character in record
E1B2	FNDLST	find last character in record
E1CB	SSEND	position side sector & buftab to end of last record
E202	BREAK	illegal system track or sector error encountered
E207	RECORD	position relative pointers to given record number or last record
		if out of range
E275	POSITN	position relative data block into active buffer & next block into
		inactive buffer
E29C	POSBUF	position proper data blocks into buffers
E2D0	BHERE	check if required block is in buffer
E2E2	NULBUF	set null records in active buffer for extension
E304	ADDNR	add record size with next record & leave in accumulator,
		if c = 1 then buffer boundary has been crossed
E31C	ADDREL	add blocks to relative file
E33B	ADDR1	— " —
E44E	NEWSS	generate new side sector and fix old side sectors to reflect it
E4FC	ERRTAB	error messages table
E60A	ETEND	end of error table
E60A	ERROR	controller error entry
E645	CMDER2	command error
E680	TLKERR	talker error recovery
E688	LSNERR	listener error recovery
E69B	HEXDEC	convert hex to bcd
E6AB	BCDDEC	convert bcd to decimal
E6BC	OKERR	transfer error message to error buffer
E706	ERMOVE	move error message from errtab to errbuf
E767	EADV1	error advance and check
E77F	BOOT2	Utility Loader: used to load user programs or system utilities
		from disk and execute them.
		format: print#15, "&0:filename"
		where file type of filename is 'usr'
		hardware required: connect data and clock line to ground.
		(2–4–5 on connector)
		on entry: only requirement is that the filename of the file to be
		loaded be the first specified name in the command
		buffer (cmdbuf) : registers – ignored
		on exit: if the file existed on disk and could be found, and no
		checksum errors were encountered, it is now loaded into
		memory, ready to execute : registers – all destroyed
		execution of the program is started at the first byte loaded
		cmdbuf contains the parameter string for the freshly loaded
		utility or user program
E7A3	UTLODR	utility loader entry point
E7D8	UTLD10	file record fetch loop
E802	UTLD30	byte storage loop
E839		Local routines used by UTLODR
E839	GTABYT	fetches a byte from the file open on the internal channel. checks
		if this was the last byte in the file. error if it was. show a
		'pter' (premature termination error)
E84B	ADDSUM	adds up checksum into location r1. algorithm: newsum =
		oldsum + newbyte + carry. inputs: expects newbyte in
		accumulator ; outputs: r1 = newsum, accumulator is destroyed
E853	ATNIRQ	start of atn/irq routines
E85B	ATNSRV	atn service routine
E909	TALK	set talk along bus
E99C	DATHI	set data out high
E9A5	DATLOW	set data out low
E9AE	CLKLOW	set clock out low
E9B7	CLKHI	set clock out high
E9C0	DEBNC	wait for reply from bus
E9C9	ACPTR	serial bus listen rtns
EA2E	LISTEN	main set listen rtn
EA4E	FRMERR	release all bus lines
EA56	ATNLOW	service atn request
EA59	TSTATN	test for atn
EA6E	PEZRO	error display routine. blinks the error# + 1 in all three leds
EAA0	DSKINT	initialize disk for routine below
EAAC	PU10	power up diagnostic
EAC9	RM10	test two 64k bit roms ; enter x = start page, exit if ok
EAEA	CR20	test all common ram
EAF0	RAMTST	test ram
EB1F	PERR2	error
EB22	DIAGOK	diagnostics ok so far
EB4B	INTTAB	initialize buffer pointer table
EBD5	SETERR	set up power on error message '73 cbm dos v2.6 1541 0 0'
EBE7	IDLE	idle loop. waiting for something to do
EC9E	STDIR	directory loading function, get the buffer and get it started
ED59	MOVBUF	transfer file name to listing buffer
ED67	GETDIR	get character for directory loading
ED84	VERDIR	validate files with bam, create new bam according to contents
		of files entered in directory
EDE5	VMKBAM	mark bam with file sectors
EE0D	NEW	new (format) a diskette
EEB7	NEWMAP	build a new map on diskette
EEF4	MAPOUT	write out the bit map to the drive in lstjob (active)
EEFF	SCRBAM	write bam according to drvnum
EF3A	SETBPT	set bit map pointer, read in bam if necessary
EF4D	NUMFRE	calculate the number of free blocks on drvnum
EF5C	WFREE	mark a track, sector as free in bam
EF88	DTYBAM	set dirty flag
EF90	WUSED	mark track, sector, (bmptr) as used
EFCF	FREUSE	calculate index into bam for target sector
EFE9	BMASK	bit mask table .byt 1,2,4,8,16,32,64,128
EFF1	FIXBAM	write the bam according to wbam flag
F005	CLRBAM	clear the bam area
F011	SETBAM	set bam image in memory, save temps
F05B	SWAP	swap images of bam
F07F	SWAP3	transfer bam to memory image
F09F	SWAP4	set pending write flag
F0A5	PUTBAM	put memory image to bam
F0BE	SWAP1	transfer image to bam
F0D1	CLNBAM	clean track number for images
F0DF	REDBAM	read in bam if not present
F10F	BAM2A	set bam pointers in buf0 & buf1 tables
F11E	NXTTS	next track & sector, returns next avaialable track & sector given
		current track & sector; allocation is from track 18 towards
		1 & 35 by full tracks
F15A	NXTERR	disk full error
F173	FNDNXT	find next optimum sector
F1A9	INTTS	returns optimum initial track & sector
F1E6	FNDSEC	— " —
F1F5	DERR	directory error
F1FA	GETSEC	set bam and find available sector starting at sector
F220	AVCK	bit map validity check
F24B	MAXSEC	number of sectors located on specific track
F258	KILLP	kill protection : not implemented with 1541 (rts)
F259	CNTINT	initialization of controller
F2B0	LCC	main controller loop. scans job que for jobs, finds job on current
		track if it exists
F2BE	TOP	top of loop to scan job que
F2F9	QUE	set status stepping
F33C	GOTU	set track & sector
F367	EXE	execute buffer
F37C	BMP	bump buffer
F393	SETJB	set job
F3B1	SEAK	scan job que
F418	DONE	return ok code
F41B	BADID	id mismatch error
F41E	CSERR	checksum error in header
F423	WSECT	find best sector to service
F497	CNVBIN	convert header in stab to binary in header
F4CA	REED	read in track, sector specified in header
F50A	DSTRT	find header and start reading data
F510	SRCH	find header
F553	ERR	header not found erro
F556	SYNC	wait for sync mark
F56E	WRIGHT	write job : write out data buffer
F5E9	CHKBLK	check block
F5F2	WTOBIN	write binary data
F691	VRFY	verify data block; convert to gcr verify image. test against data
		block, convert back to binary
F6CA	SECTSK	sector seek
F6D0	PUT4BG	fast binary to gcr routine
F77F	BGTAB	binary to gcr table
		; .byt $0A,$0B,$12,$13,$0E,$0F,$16,$17
		; .byt $09,$19,$1A,$1B,$0D,$1D,$1E,$15
F78F	BINGCR	binary to gcr conversion routine; does in place conversion of
		buffer to gcr using overflow block
		creates write image: 1 block id character + 256 data bytes + 1
		check sum + 2 off bytes = 260 binary bytes
		260 binary bytes = 325 gcr
		325 = 256 + 69 overflow (ie. 10 bits/gcr byte)
F7E6	GET4GB	fast gcr to binary conversion
F8A0	GCRHI	gcr high byte table
		an occurance of $ff in either the high or low table constitutes an
		error
		.byt $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
		.byt $FF,$80,$00,$10,$FF,$C0,$40,$50
		.byt $FF,$FF,$20,$30,$FF,$F0,$60,$70
		.byt $FF,$90,$A0,$B0,$FF,$D0,$E0,$FF
F8C0	GCRLO	gcr low byte table
		.byt $FF,$FF,$FF,$FF,$FF,$FF,$FF,$FF
		.byt $FF,$08,$00,$01,$FF,$0C,$04,$05
		.byt $FF,$FF,$02,$03,$FF,$0F,$06,$07
		.byt $FF,$09,$0A,$0B,$FF,$0D,$0E,$FF
F8E0	GCRBIN	entry point for get4gb, fast gcr to binary conversion
F934	CONHDR	convert header into gcr search image and place in stab image
		contains 00 1D 1D TR SC SC HBID
F969	ERRR	disk controller error handling rtn
F97E	TURNON	turn on drive motor
F98F	TRNOFF	turn off drive motor
F99C	END	motor and stepper control routines
F99C	END	irq into controller every 10 ms
F9A5	END001	test write protect
F9B1	END002	test for phase offset
F9CB	END40	no phase offset, check active drive
F9D6	END33X	nothing happening, exit stage left to end33
F9D9	END10	something is happening, check what it is
F9FA	END30	test if step needed
FA02	END30X	goto proper stepper state jmp (nxtst)
FA05	INACT	get abs(steps)
FA0E	INAC10	test if we can accel, set up short step mode
FA1C	INAC20	calculate & set up number of accel steps
FA2E	DOSTEP	do step, check if in or out
FA32	STPOUT	step out
FA3B	SHORT	short step mode, set up settle pointers in nxtst and settle time
		in aclstp
FA4E	SETLE	decrement settle time factor, set up inactive state (inact)
		pointers in nxtst
FA63	STPIN	step in
FA7B	SSACL	sub acel factor
FA97	SSRUN	fast stepping mode rtn
FAA5	SSDEC	sub decel factor
FABE	END33	disable s.o. to 6502
FAC7	CODE	format routine for lcc ($F2B0)
FAF5	L213	test if on right track number
FB00	L214	test for write protect
FB39	FWAIT	wait for sync
FB3E	FWAIT2	wait for no sync
FB43	F000	reset interrupt flags (ifr) to start timer
FB7D	F009	calculate the differance between sync and nonsync and adjust
		num accordingly
FBB6	COUNT	count number of bytes in data segment
FBCE	CNT20	sync found
FBE0	DS08	calc number bytes required to record on track
FC3F	MAK10	create header images
FC88	CRTDAT	create data block of zero
FCB1	WRTSYN	write sync
FD2C	COMP	verify format
FD39	CMPR10	find sync
FD40	CMPR15	get header byte
FD58	CMPR20	test if too many errors
FD62	TSTDAT	test data
FD96	FMTEND	format rtn end
FDA3	SYNCLR	erase track with sync
FDC3	WRTNUM	write out num syncs
FDD3	FMTERR	format error
FDE5	MOVUP	move number of bytes in buffer up 69 bytes
FE00	KILL	disable write
FE0E	CLEAR	enable write
FE30	FBTOG	format binary to gcr conversion, converts buffer to gcr without
		hbid
FE67	SYSIRQ	main system irq routine
FE85	DIRTRK	.byt 18 : directory track number
FE86	BAMSIZ	.byt 4 : number of bytes/track in bam
FE87	MAPOFF	.byt 4 : offset of bam in sector
FE88	DSKNAM	.byt $90 : offset of disk name in bam sector
FE89	CMDTBL	.byt 'vidmbup&crsn' : command search table
		(validate, initialize. duplicate, m– , b– , user,
		position, utlodr, copy, rename, scratch, new)
FE95	CJUMPL	command jump table low
FEA1	CJUMPH	command jump table high
FEA5	STRUCT	structure images for commands
		.byt %01010001 disk copy
		.byt %11011101 rename
		.byt %00011100 scratch
		.byt %10011110 new
		.byt %00011100 load
FEB2	MODLST	.byt 'rwam' mode table
FEB6	TPLST	.byt 'dspul' file type table (del, seq, prg, usr, rel)
FEBB	TYPLST	.byt 'dspur'
FEC0	TP1LST	.byt 'eerse'
FEC5	TP2LST	.byt 'lqgrl'
FECA	LEDMSK	led mask
FECC	ER00	error flag variables for bit
FECC	ER00	.byt $00
FECD	ERO	.byt $3F
FECE	ER1	.byt $7F
FECF	ER2	.byt $BF
FED0	ER3	.byt $FF
FED1	NUMSEC	.byt 17,18,19,21 ; number of differant sectors/track
FED5	VERNUM	.byt 41 ; format type
FED6	NZONES	.byt 4 ; number of density zones
FED7	MAXTRK	.byt 36 ; maximum track number + 1
FED7	TRKNUM	.byt 31,25,18 ; zone boundaries track numbers
FEDB	OFFSET	.byt 1,$FF,$FF,1,0 ; offset for recovery
FEE0	BUFIND	.byt $03,$04,$05,$06,$07,$07 ; buffer index
FEE6	ECHKSM	.byt 0 ; $E–$F rom checksum
FEE7	NMI	jmp (vnmi) : $0065
FEEA	PEA7A	patch for power on errors
FEF3	SLOWD	patch area for 1541 disk with slow serial receive
FEFB	CLKDAT	new clock low and data high for patch area
FF01	NNMI	new nmi routine
FFE6		default table for user command
		$C8C6 format
		$F98F trnoff
FFEA	UBLOCK	user command set up
		$CD5F UBLKRD user block read
		$CD97 UBLKWT user block write
		$0500 ——— links to buffer #2
		$0503 ——— — " —
		$0506 ——— — " —
		$0509 ——— — " —
		$050C ——— — " —
		$050F ——— — " —
		$FF01 NNMI new nmi routine
		$EAA0 DSKINT disk initialization
		$FE67 SYSIRQ

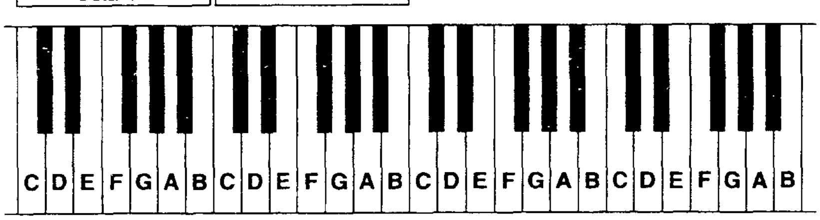
Music Symbols

8va	Above staff: play 1 octave higher (Note = Note x 2) Below staff: play 1 octave lower (Note = Note / 2)
	Slur or Bowing: Indicates *Legato* when connecting a group of notes. Indicates a *Tie* when connecting 2 notes of the same pitch (2nd note is NOT played - value of 2nd note is added to the value of the 1st note).
tr	**Trill:** Alternate adjacent notes rapidly.
or	**Mordent:** Play note, add next higher note and release, holding 1st note.
	Inverted Mordent: Play note, add next lower note and release, holding 1st note.
	Pedal: Attack and Release.
	Pedal Release.
	Turn.
	Dal Segno: Like GOTO (label).
	Crescendo: Smoothly increasing intensity.
	Decrescendo: Smoothly decreasing intensity.
1. 2.	**First & Second Endings:** Play ending 1, then 2 (omit 1)
	Repeat Marks: Like FOR I = 1 TO 2.
	Repeat Measure.
	Fermata or Hold.
	Indicates voice line moving from one staff to another.
	Arpeggiate: Play notes in a chord successively from bottom to top, or top to bottom, respectively.
gliss.	**Glissando:** Slide notes.
	Down-Bow, Up-Bow: For stringed instruments.
> ∧	**Accent Marks:** Intensity or pressure increase on note.

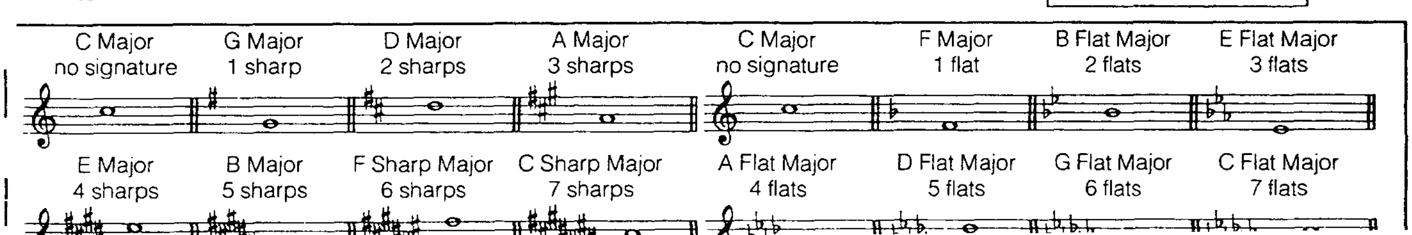
	Slight Accent.
	Staccato Marks: Shorten duration of note(s).
	Moderate Staccato.
♩ = 120	**Metronome Setting.**
	Clefs: Treble or G, Bass or F, C Clef.
3 4 5	**Beat Interrupts:** Divide the beat into other than the regular notation.
♯ ×	**Sharp, Double Sharp.**
♭ ♭♭	**Flat, Double Flat.**
♮	**Natural.**
2/4 6/8 3/2 C ¢	**Meter Signatures:** 2/4, 6/8, 3/2, 4/4, 2/2, respectively.
	Whole Rest, Half Rest, Quarter Rest.
	1/8 Rest, 1/16 Rest, 1/32 Rest.
4	**Multiple Measure Rest:** Rest for n measures.
	Natural Harmonic: On stringed instruments.
	Artificial Harmonic on the Violin. Sounds 2 octaves above lower tone.
	Notes: Double Whole (breve), Whole (semibreve), Half (minim), Quarter (crotchet).
	Notes: Eighth (quaver), Sixteenth (semiquaver), Thirty-Second (demisemiquaver).
	Dotted Note: Increment duration by 50%.
	Tremolo: Repeat rapidly for duration of note.







C	Doh	Tonic
D	Ray	Supertonic
E	Me	Mediant
F	Fah	Subdominant
G	Soh	Dominant
A	Lah	Submediant
B	Te	Leading Note
C	Doh	Tonic



Note Frequency Table

Frequency in Hz

Based on formula: $Note_N = Note_{N-1} \times 2 \uparrow (1/12)$

(– Octave Not Accessible) (* Octave Only Partially Accessible)

Note in:	Octave:								
For:	0	1	2	3	4	5	6	7	8
CB2	–	–	–	–	0	1	2	3	–
VIC Voice 1	–	0	1	2	3*	–	–	–	–
VIC Voice 2	–	–	0	1	2	3*	–	–	–
VIC Voice 3	–	–	–	0	1	2	3*	–	–
C64	0	1	2	3	4	5	6	7	–
+4/C16	–	–	0	1	2	3	4	5	6
C	16.3516	32.7032	65.4064	130.813	261.626	523.251	1046.50	2093.00	4186.01
C#	17.3239	34.6478	69.2957	138.591	277.183	554.365	1108.73	2217.46	4434.92
D	18.3540	36.7081	73.4162	146.832	293.665	587.330	1174.66	2349.32	4698.64
D#	19.4454	38.8909	77.7817	155.563	311.127	622.254	1244.51	2489.02	4978.03
E	20.6017	41.2034	82.4069	164.814	329.628	659.255	1318.51	2637.02	5274.04
F	21.8268	43.6536	83.3071	174.614	349.228	698.456	1696.91	2793.83	5587.65
F#	23.1247	46.2493	92.4986	184.997	369.994	739.989	1479.98	2959.96	5919.91
G	24.4997	48.9994	97.9989	195.998	391.995	783.991	1567.98	3135.96	6271.93
G#	25.9565	51.9131	103.826	207.652	415.305	830.609	1661.22	3322.44	6644.88
A	27.5	55.0	110.0	220.0	440.0	880.0	1760.0	3520.0	7040.0
A#	29.1352	58.2705	116.541	233.082	466.164	932.328	1864.66	3729.31	7458.62
B	30.8671	63.7354	123.471	246.942	493.883	987.767	1975.53	3951.07	7902.13

Chord Note Derivatives

Notes are shown in diminishing order of importance.

Chord	Major	Minor	Seventh	Minor 7th	Diminished
A♭ / G#	A♭ C E♭	G# B D#	A♭ C G♭ E♭	G# B F# D#	G# B D F
A	A C# E	A C E	A C# G E	A C G E	A C E♭ F#
B♭ / A#	B♭ D F	B♭ D♭ F	B♭ D A♭ F	B♭ D♭ A♭ F	B♭ D♭ E G
B / C♭	B D# F#	B D F#	B D# A F#	B D A F#	B D F A♭
C / B#	C E G	C E♭ G	C E B♭ G	C E♭ B♭ G	C E♭ F# A
D♭ / C#	D♭ F A♭	C# E G#	D♭ F C♭ A♭	C# E B G#	C# E G A#
D	D F# A	D F A	D F# C A	D F C A	D F A♭ B
E♭ / D#	E♭ G B♭	E♭ G♭ B♭	E♭ G D♭ B♭	E♭ G♭ D♭ B♭	E♭ G♭ A C
E / F♭	E G# B	E G B	E G# D B	E G D B	E G B♭ D♭
F / E#	F A C	F A♭ C	F A E♭ C	F A♭ E♭ C	F A♭ B D
E♭ / F#	F# A# C#	F# A C#	F# A# E C#	F# A# E C#	F# A C D#
G	G B D	G B♭ D	G B F D	G B♭ F D	G B♭ D♭ E

Chord	Augmented	Suspended 4th	Major 7th	Major 6th	Major 9th
A♭ / G#	A♭ C E	A♭ D♭ E♭	A♭ C G E♭	A♭ C F E♭	A♭ C B♭ G♭ E♭
A	A C# F	A D E	A C# G# E	A C# F# E	A C# B♭ G♭ E♭
B♭ / A#	B♭ D F#	B♭ E♭ F	B♭ D A F	B♭ D G F	B♭ D C A♭ F
B / C♭	B D# G	B E F#	B D# A# F#	B D# G# F#	B D# C# A F#
C / B#	C E G#	C F G	C E B G	C E A G	C E D B♭ G
D♭ / C#	D♭ F A	D♭ G♭ A♭	D♭ F C A♭	D♭ F B♭ A♭	D♭ F E♭ C♭ A♭
D	D F# A#	D G A	D F# C# A	D F# B A	D F# E C A
E♭ / D#	E♭ G B	E♭ A♭ B♭	E♭ G D B♭	E♭ G C B♭	E♭ G F D♭ B♭
E / F♭	E G# C	E A B	E G# D# B	E G# C# B	E G# F# D B
F / E#	F A C#	F B♭ C	F A E C	F A D C	F A G E♭ C
E♭ / F#	F# A# D	F# B C#	G♭ B♭ F D♭	G# A# D# C#	F# A# G# E C#
G	G B D#	G C D	G B F# D	G B E D	G B A F D

CB2 Note Values

Reset Port with POKE (PET:59467 / VIC:37147 / C64:56587), 0

PET/CBM : POKE 59467,16 : POKE 59466, (Oct) : POKE 59464, X
VIC 20 : POKE 37147,16 : POKE 37146, (Oct) : POKE 37144, X
C64 : POKE 56587,16 : POKE 56586, (Oct) : POKE 56584, X

Note	Oct = 15		Oct = 51		Oct = 85	
	Octave 0	Octave 1	Octave 1	Octave 2	Octave 2	Octave 3
B	251b	125	251	125	251	125
C	238	118	238	118	238	118
C#	224	110	224	110	224	110
D	210	104	210	104	210	104
D#	199	99	199	99	199	99
E	188	93	188	93	188	93
F	177	88	177	88	177	88
F#	168	83	168	83	168	83
G	158	78	158	78	158	78
G#	149	74	149	74	149	74
A	140	69	140	69	140	69
A#	133	65	133	65	133	65

Square Wave Frequency Formulae: where: Clock = 1,000,000

Frequency Output (F) = Clock / 2 (N + 2) (C)
Number in Table (N) = (Clock / F x C x 2) – 2

C = 8 for **Oct = 15**
C = 4 for **Oct = 51**
C = 2 for **Oct = 85**

VIC 20 Note Values

Where two values are shown,
it is necessary to alternate between them to get the true note.

Voice frequency registers are 36874/5/6. • Noise reg is 36877.
Volume is Lo nybble of 36878. See Memory Map

Note	Octave 0		Octave 1		Octave 2		Octave 3	
	Value	Alt.	Value	Alt.	Value	Alt.	Value	Alt.
C	131		192	195	224		239	240
C#	140		197		226		240	241
D	145		200		227	228		
D#	151		203		229			
E	158		206	207	231			
F	161	162	208	209	232			
F#	166	167	211	212	233			
G	173	174	214		234	235		
G#	178		216		238	236		
A	181	182	218	219	237			
A#	185	186	220	221	237	238		
B	189	190	222	223	239			

VIC Chip Frequency Formulae:

Frequency Output (F) = Clock / (255 – N)
Number in Table (N) = 255 – (Clock/F)

	NTSC (N.America)	PAL (European)
VIC 20 Voice 1 (36874): Clock =	3995	4329
VIC 20 Voice 2 (36875): Clock =	7990	8659
VIC 20 Voice 3 (36876): Clock =	15980	17320
VIC 20 Voice 4 (36877): Clock =	31960	34640

Commodore 64 SID Note Values

The value under Hi is POKEd into the Hi byte of the frequency registers (54273, 54280, 54287). Likewise with Lo (54272, 54279, 54286)

Note	Octave 0			Octave 1			Octave 2			Octave 3		
	Oscillator Frequency			Oscillator Frequency			Oscillator Frequency			Oscillator Frequency		
	Decimal =	Hi (x256) +	Lo	Decimal =	Hi (x256) +	Lo	Decimal =	Hi (x256) +	Lo	Decimal =	Hi (x256) +	Lo
C	268	1	12	536	2	24	1072	4	48	2145	8	97
C#	284	1	28	568	2	56	1136	4	112	2273	8	225
D	301	1	45	602	2	90	1204	4	180	2408	9	104
D#	318	1	62	637	2	125	1275	4	251	2551	9	247
E	337	1	81	675	2	163	1351	5	71	2703	10	143
F	358	1	102	716	2	204	1432	5	152	2864	11	48
F#	379	1	123	758	2	246	1517	5	237	3034	11	218
G	401	1	145	803	3	35	1607	6	71	3215	12	143
G#	425	1	169	851	3	83	1703	6	167	3406	13	78
A	451	1	195	902	3	134	1804	7	12	3608	14	24
A#	477	1	221	955	3	187	1911	7	119	3823	14	239
B	506	1	250	1012	3	244	2025	7	233	4050	15	210

NTSC: Frequency Out = Note Value / 16.40426
Note Value = Frequency Out x 16.40426

PAL: Frequency Out = Note Value / 17.77984
Note Value = Frequency Out x 17.77984

Note	Octave 4			Octave 5			Octave 6			Octave 7		
	Oscillator Frequency			Oscillator Frequency			Oscillator Frequency			Oscillator Frequency		
	Decimal =	Hi (x256) +	Lo	Decimal =	Hi (x256) +	Lo	Decimal =	Hi (x256) +	Lo	Decimal =	Hi (x256) +	Lo
C	4291	16	195	8583	33	135	17167	67	15	34334	134	30
C#	4547	17	195	9094	35	134	18188	71	12	36376	142	24
D	4817	18	209	9634	37	162	19269	75	69	38539	150	139
D#	5103	19	239	10207	39	223	20415	79	191	40830	159	126
E	5407	21	31	10814	42	62	21629	84	125	43258	168	250
F	5728	22	96	11457	44	193	22915	89	131	45830	179	6
F#	6069	23	181	12139	47	107	24278	94	214	48556	189	172
G	6430	25	30	12860	50	60	25721	100	121	51443	200	243
G#	6812	26	156	13625	53	57	27251	106	115	54502	212	230
A	7217	28	49	14435	56	99	28871	112	199	57743	225	143
A#	7647	29	223	15294	59	190	30588	119	124	61176	238	248
B	8101	31	165	16203	63	75	32407	126	151	64814	253	46

Commodore 64 SID Envelope Rates

Master Volume (MV) = Lo nybble of 54296. MV & ADSR Regs (R1 & R2) are write only.
Voice1: 54277/8 • Voice2: 54284/5 • Voice3: 54291/2. See Memory Map.

Value		POKE R1, (Hi + Lo)		POKE R2, (Hi + Lo)	
		Hi nybble	Lo nybble	Hi nybble	Lo nybble
Hi nybble	Lo nybble	Attack Rate 0 to peak	Decay Rate peak to SL	Sustain Level Val/15th's of MV	Release rate SL to 0
0	0	2 ms	6 ms	0/15MV	6 ms
16	1	8 ms	24 ms	1/15MV	24 ms
32	2	16 ms	48 ms	2/15MV	48 ms
48	3	24 ms	72 ms	3/15MV	72 ms
64	4	38 ms	114 ms	4/15MV	114 ms
80	5	56 ms	168 ms	5/15MV	168 ms
96	6	68 ms	204 ms	6/15MV	204 ms
112	7	80 ms	240 ms	7/15MV	240 ms
128	8	100 ms	300 ms	8/15MV	300 ms
144	9	250 ms	750 ms	9/15MV	750 ms
160	10	500 ms	1.5 s	10/15MV	1.5 s
176	11	800 ms	2.4 s	11/15MV	2.4 s
192	12	1.0 s	3.0 s	12/15MV	3.0 s
208	13	3.0 s	9.0 s	13/15MV	9.0 s
224	14	5.0 s	15.0 s	14/15MV	15.0 s

+4 / C16 Sound

The numbers in the table are used as the second parameter of the SOUND command.

Note	Octave 0	Octave 1	Octave 2	Octave 3	Octave 4
A	7	516	770	897	960
A#	64	544	784	904	964
B	118	571	798	911	967
C	169	596	810	917	970
C#	224	620	822	923	974
D	262	643	834	929	976
D#	305	664	844	934	979
E	345	685	854	939	981
F	383	704	864	944	984
F#	419	721	872	948	986
G	453	739	881	953	988
G#	485	754	889	956	990

+4 / C16 Frequency Formulae:

NTSC:
Frequency Output = 111860.781 / (1024 – SOUND Value)
SOUND Value = 1024 – (111860.781 / Frequency Output)

PAL:
Frequency Output = 111840.450 / (1024 – SOUND Value)
SOUND Value = 1024 – (111840.450 / Frequency Output)

VIC 20 Screen & Border Colours

POKE 36879, X :				Border				
Screen	BLK	WHT	RED	CYN	PUR	GRN	BLU	YEL
BLK	8	9	10	11	12	13	14	15
WHT	24	25	26	27	28	29	30	31
RED	40	41	42	43	44	45	46	47
CYN	56	57	58	59	60	61	62	63
PUR	72	73	74	75	76	77	78	79
GRN	88	89	90	91	92	93	94	95
BLU	104	105	106	107	108	109	110	111
YEL	120	121	122	123	124	125	126	127
ORG	136	137	138	139	140	141	142	143
Lt. ORG	152	153	154	155	156	157	158	159
PNK	168	169	170	171	172	173	174	175
Lt. CYN	184	185	186	187	188	189	190	191
Lt. PUR	200	201	202	203	204	205	206	207
Lt. GRN	216	217	218	219	220	221	222	223
Lt. BLU	232	233	234	235	236	237	238	239
Lt. YEL	248	249	250	251	252	253	254	255

Colour Codes

Colour:	VIC	C64	+4	ASCII	Colour:	VIC	C64	+4	ASCII
Black	0	0	1	144	Medium Grey		12		152
White	1	1	2	5	Light Purple	12*			
Red	2	2	3	28	Blue-Green			13	152
Cyan	3	3	4	159	Light Green	13*	13	16	153
Purple	4	4	5	156	Light Blue	14*	14	14	154
Green	5	5	6	30	Dark Blue			15	154
Blue	6	6	7	31	Light Grey		15		155
Yellow	7	7	8	158	Light Yellow	15*			
Orange	8*	8	9	129					
Brown		9	10	149					
Light Orange	9*								
Pink	10*	10	12	150					
Yellow-Green			11	150					
Dark Grey		11		151					
Light Cyan	11*								

* = Not available as a character colour. Colour values for VIC/C64 are POKEd into the appropriate registers (see memory maps). +4 values are used in the COLOR Command (same for C16). ASCII values are PRINTed using CHR$.

Commodore 6545 Video Chip

POKE 59520, R#	POKE 59521, Value
R0	Horizontal total number of characters on line (Nht) including horizontal retrace. (true value = number + 1)
R1	Horizontal number of characters displayed (Nhd)
R2	Distance (in characters) from left to right margin of screen + 1
R3	Sync width. Lo nybble is vertical sync width (in lines) Hi nybble is horizontal sync (in characters).
R4	Number of display lines including retrace (Nvt).
R5	Vertical position of the edge of the screen.
R6	Number of display lines on screen (Nvd)
R7	Height of upper edge from bottom of screen (in lines displayed)
R8	Interlace and Skew:– Bit 0 1 = interlaced mode 0 = non interlaced mode Bit 1 if Bit 0 = 1 then interlace and video mode Bit 2 not used Bit 3 not used Bit 4 1 = scan from 32770 in memory Bit 5 1 = scan from 32772 in memory Bit 6 cursor (not implemented on the PET) Bit 7 cursor (not implemented on the PET)
R9	Number of lines between top of one display line and top of the next
R10	Cursor (not implemented on the PET)
R11	Cursor (not implemented on the PET)
R12	Control Register: Bit 0 add 256 to start address (512 for 8032) Bit 1 add 512 to start address (1024 for 8032) Bit 2 invert flyback Bit 3 invert video signal Bit 4 use top half of 4K character generator Bit 5 (not implemented on the PET) Bit 6 (not implemented on the PET) Bit 7 not used
R13	Value + 32768 is address of first character (multiply by 2 for 8032)
R14	Cursor location HI (not implemented on the PET)
R15	Cursor location LO (not implemented on the PET)
R16	Light pen position HI (read only)
R17	Light pen position LO (read only)

Table Of Secondary Addresses

Eg. OPEN 4, 4, 7 : 7 is the Secondary Address on CBM printers that alters line spacing. Once open the new value can be sent. Secondary addresses are not applicable to the VIC 20/Commodore 64 RS-232 routines ('device' 2), keyboard (device 0), screen (device 3), or the CBM 8010 Modem (device 5).

	I/O Device & Device Number (DV#)			
Sec. Addr.	Printer 4	Cassette 1 or 2	Vic/64 Cassette 1	Disk 8
0	Print data exactly as received	seq. read	Load & relocate (dflt)	Load, and Dir read
1	Print data according to previously defined format	Write file + end-of-file marker on Close	Load without relocating	Program Save
2	Format Set-up	Write file + eof + end of tape marker on Close	Write file + eof + end of tape marker on Close	R/W channels are 2–14
3	Set number of lines per page for paging			
4	Enable printer format diagnostics			
5	Define a programmable character			
6	Set spacing between lines			
7	Upper/Lower case			
8	ASCII/Graphics			
9	Suppress Diagnostic Message Printing			
10	Reset Printer			
11	Set Uni-Direction			
12	Reset Uni-Direction			
13	Set Condense mode			
14	Reset Condense mode			
15	Set pseudo letter quality			Command Ch.
21	Reset pseudo letter quality			
17	Storing bit image data			
18	Printing bit data previously written			

8032 Control Characters

Most functions can be activated by combinations of simultaneous key depressions, a phenomena of the keyboard hardware. Notice that the CHR$ values of complimentary functions differ by 128.

Function	CHR$	ESC/RVS	Keyboard Combination
BELL	7	G	
GRAPHICS TEXT	142 14	Shift N N	Both Shifts + *
SCROLL DOWN SCROLL UP	153 25	Shift Y Y	Left Shift + TAB + I
SET BOTTOM SET TOP	143 15	Shift O O	Shift + Z + A + L Z + A + L
INSERT LINE DELETE LINE	149 21	Shift U U	Shift + RVS + A + L RVS + A + L
ERASE BEGIN ERASE END	150 22	Shift V V	Shift + TAB + ⊟ + DEL TAB + ⊟ + DEL
SET/CLR TAB TAB	137 9	Shift I I	Shift + TAB TAB

8032 Window POKEs	
TOP:224, T where T = 0 to 24	LEFT:226, L where L = 0 to 79
BOTTOM:225, B where B = T to 24	RIGHT:213, R where R = L to 79

VIC 20 Screen Memory

To move the screen: POKE 36869, (PEEK(36869) AND 15) OR X
POKE 36866, (PEEK(36866) AND 127) OR Y

X	Y	4•(PEEK(36866) AND 128) + 64•(PEEK(36869) AND 112) = Location	
		Decimal (½K blocks)	Hexadecimal
128	0	0	$0000
128	128	512	$0200
129	0	1024	0400
129	128	1536	0600
130	0	2048	0800
130	128	2560	0A00
131	0	3072	0C00
131	128	3584	0E00
132	0	4096	1000 (dflt w/exp)
132	128	4608	1200
133	0	5120	1400
133	128	5632	1600
134	0	6144	1800
134	128	6656	1A00
135	0	7168	1C00
135	128	7680	1E00 (default)
136	0	8192	2000
136	128	8704	2200
137	0	9216	2400
137	128	9728	2600
138	0	10240	2800
138	128	10752	2A00
139	0	11264	2C00
139	128	11776	2E00
140	0	12288	3000
140	128	12800	3200
141	0	13312	3400
141	128	13824	3600
142	0	14336	3800
142	128	14848	3A00
143	0	15360	3C00
143	128	15872	3E00

Commodore 64 Screen Memory

To move the screen: POKE 53272, (PEEK(53272)AND15) OR X

X	(3–PEEK(56576) AND 3) • 16384 + (X•64) = Location For Screen at Bank 0 (default):	
	Decimal	Hexadecimal
0	0	$0000
16	1024	0400 (default)
32	2048	0800
48	3072	0C00
64	4096	1000
80	5120	1400
96	6144	1800
112	7168	1C00
128	8192	2000
144	9216	2400
160	10240	2800
176	11264	2C00
192	12288	3000
208	13312	3400
224	14336	3800
240	15360	3C00

Commodore 64 VIC II Address

To move VIC II: POKE 56576, (PEEK(56576) AND 252) OR X ;X=3–Bank#

Bank	X	VIC II Chip Address Range	
		Decimal (16K blocks)	Hexadecimal
0	3	0–16383	$0000–3FFF (default)
1	2	16384–32767	4000–7FFF
2	1	32768–49151	8000–BFFF
3	0	49152–65535	C000–FFFF

Note: Character ROM only available with VIC II in bank 0 or 2

VIC 20 Character Base

To move the character base: POKE 36869, (PEEK(36869) AND 240) OR X

X*	32768 + (PEEK(36869) AND 15) • 1024 = Location	
	Decimal (1K blocks)	Hexadecimal
0	32768–34815	$8000–87FF (dflt)
1	33792–35839	8400–8BFF
2	34816–36863	8800–8FFF
3	35840–37887	8C00–93FF
4	36864–38911	9000–97FF
5	37888–39935	9400–9BFF
6	38912–40959	9800–9FFF
7	39936–41983	9C00–A3FF
8	0–2047	0000–07FF
9	1024–3071	0400–0BFF
10	2048–4095	0800–0FFF
11	3072–5019	0C00–13FF
12	4096–6143	1000–17FF
13	5020–7167	1400–1BFF
14	6144–8191	1800–1FFF
15	7168–9216	1C00–23FF

* X=PEEK(36869) AND 15

Commodore 64 Character Base

To move the character base: POKE 53272, (PEEK(53272) AND 240) OR X

X*	(3–PEEK(56576) AND 3) • 16384 + (X•64) = Location For Screen at Bank 0 (default):	
	Decimal (2K blocks)	Hexadecimal
0	0–2047	$0000–07FF
2	2048–4095	0800–0FFF
4	4096–6143	1000–17FF *1
6	6144–8191	1800–1FFF *2
8	8192–10293	2000–27FF
10	10240–12287	2800–2FFF
12	12288–14335	3000–37FF
14	14336–16383	3800–3FFF

* – X=PEEK(53272) AND 14
*1 – Lower 2K of Character ROM (Bank 0 or 2 only) (default)
*2 – Upper 2K of Character ROM (Bank 0 or 2 only)

Character ROM Contents

Character ROM is the same in all machines, but only addressable in VIC 20/C64

2K Block	VIC 20 Default Address		Commodore 64 Default Address		Commodore 64 VIC II Image	Contents
	Dec (½K blocks)	Hex	Dec (½K blocks)	Hex	Hex	
0	32768–33279	8000–81FF	53248–53759	D000–D1FF	1000–11FF	Upper case characters
	33280–33791	8200–83FF	53760–54271	D200–D3FF	1200–13FF	Graphics characters
	33792–34303	8400–85FF	54272–54783	D400–D5FF	1400–15FF	Reversed upper case characters
	34304–34815	8600–87FF	54784–55295	D600–D7FF	1600–17FF	Reversed graphics characters
1	34816–35327	8800–89FF	55296–55807	D800–D9FF	1800–19FF	Lower case characters
	35328–35839	8A00–8BFF	55808–56319	DA00–DBFF	1A00–1BFF	Upper case and graphics characters
	35840–36351	8C00–8DFF	56320–56831	DC00–DDFF	1C00–1DFF	Reversed lower case characters
	36352–36863	8E00–8FFF	56832–57343	DE00–DFFF	1E00–1FFF	Reversed upper case and graphics

Sprite Design

Sprite Colour #2 _____ : POKE 53285, _____
Sprite Colour #3 _____ : POKE 53286, _____
Sprite Enable: POKE 53269, PEEK(53269) OR 2 ↑ Sprite#
POKE Sprite X-Expand: POKE 53264, PEEK(53264) OR 2 ↑ Sprite#
Sprite Y-Expand: POKE 53271, PEEK(53271) OR 2 ↑ Sprite#
Background Priority: POKE 53275, PEEK(53275) OR 2 ↑ Sprite#

Sprite Multi Colour Mode: POKE 53276, PEEK(53276) OR 2 ↑ Sprite#

Multi Colour Mode Bit Pairs

Background Colour, PEEK(53281), Use: 00
Sprite Colour Use: 01
Sprite Colour #2 Use: 10
Sprite Colour #3 Use: 11

Column 1	2	3	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Column 1	2	3
0	1	2				00	01	02
3	4	5				03	04	05
6	7	8				06	07	08
9	10	11				09	0A	0B
12	13	14				0C	0D	0E
15	16	17				0F	10	11
18	19	20				12	13	14
21	22	23				15	16	17
24	25	26				18	19	1A
27	28	29				1B	1C	1D
30	31	32				1E	1F	20
33	34	35				21	22	23
36	37	38				24	25	26
39	40	41				27	28	29
42	43	44				2A	2B	2C
45	46	47				2D	2E	2F
48	49	50				30	31	32
51	52	53				33	34	35
54	55	56				36	37	38
57	58	59				39	3A	3B
60	61	62				3C	3D	3E
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			

Sprite # _____ (0–7)
Pointer: POKE 2040 + Sprite#, _____
Sprite Colour: _____ : POKE 53287 + Sprite#, _____
X-Position: POKE 53248 + Sprite#, X Position
Y-Position: POKE 53249 + Sprite#, Y Position

Column 1	2	3	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Column 1	2	3
0	1	2				00	01	02
3	4	5				03	04	05
6	7	8				06	07	08
9	10	11				09	0A	0B
12	13	14				0C	0D	0E
15	16	17				0F	10	11
18	19	20				12	13	14
21	22	23				15	16	17
24	25	26				18	19	1A
27	28	29				1B	1C	1D
30	31	32				1E	1F	20
33	34	35				21	22	23
36	37	38				24	25	26
39	40	41				27	28	29
42	43	44				2A	2B	2C
45	46	47				2D	2E	2F
48	49	50				30	31	32
51	52	53				33	34	35
54	55	56				36	37	38
57	58	59				39	3A	3B
60	61	62				3C	3D	3E
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			

Sprite # _____ (0–7)
Pointer: POKE 2040 + Sprite#, _____
Sprite Colour: _____ : POKE 53287 + Sprite#, _____
X-Position: POKE 53248 + Sprite#, X Position
Y-Position: POKE 53249 + Sprite#, Y Position

Column 1	2	3	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Column 1	2	3
0	1	2				00	01	02
3	4	5				03	04	05
6	7	8				06	07	08
9	10	11				09	0A	0B
12	13	14				0C	0D	0E
15	16	17				0F	10	11
18	19	20				12	13	14
21	22	23				15	16	17
24	25	26				18	19	1A
27	28	29				1B	1C	1D
30	31	32				1E	1F	20
33	34	35				21	22	23
36	37	38				24	25	26
39	40	41				27	28	29
42	43	44				2A	2B	2C
45	46	47				2D	2E	2F
48	49	50				30	31	32
51	52	53				33	34	35
54	55	56				36	37	38
57	58	59				39	3A	3B
60	61	62				3C	3D	3E
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			

Sprite # _____ (0–7)
Pointer: POKE 2040 + Sprite#, _____
Sprite Colour: _____ : POKE 53287 + Sprite#, _____
X-Position: POKE 53248 + Sprite#, X Position
Y-Position: POKE 53249 + Sprite#, Y Position

Column 1	2	3	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Bit 7 6 5 4 3 2 1 0	Column 1	2	3
0	1	2				00	01	02
3	4	5				03	04	05
6	7	8				06	07	08
9	10	11				09	0A	0B
12	13	14				0C	0D	0E
15	16	17				0F	10	11
18	19	20				12	13	14
21	22	23				15	16	17
24	25	26				18	19	1A
27	28	29				1B	1C	1D
30	31	32				1E	1F	20
33	34	35				21	22	23
36	37	38				24	25	26
39	40	41				27	28	29
42	43	44				2A	2B	2C
45	46	47				2D	2E	2F
48	49	50				30	31	32
51	52	53				33	34	35
54	55	56				36	37	38
57	58	59				39	3A	3B
60	61	62				3C	3D	3E
			7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0			

Sprite # _____ (0–7)
Pointer: POKE 2040 + Sprite#, _____
Sprite Colour: _____ : POKE 53287 + Sprite#, _____
X-Position: POKE 53248 + Sprite#, X Position
Y-Position: POKE 53249 + Sprite#, Y Position

Character Design

Bit

	7	6	5	4	3	2	1	0
0								
1								
2								
3								
4								
5								
6								
7								

Character # ______

Bit

	7	6	5	4	3	2	1	0
0								
1								
2								
3								
4								
5								
6								
7								

Character # ______

Bit

	7	6	5	4	3	2	1	0
0								
1								
2								
3								
4								
5								
6								
7								

Character # ______

Screen Design

40 Column PET/CBM Screen Map

32768

1	8000	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799	2800	2801	2802	2803	2804	2805	2806	2807
2	8028	2808	2809	2810	2811	2812	2813	2814	2815	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
3	8050	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879	2880	2881	2882	2883	2884	2885	2886	2887
4	8078	2888	2889	2890	2891	2892	2893	2894	2895	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
5	80A0	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959	2960	2961	2962	2963	2964	2965	2966	2967
6	80C8	2968	2969	2970	2971	2972	2973	2974	2975	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
7	80F0	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039	3040	3041	3042	3043	3044	3045	3046	3047
8	8118	3048	3049	3050	3051	3052	3053	3054	3055	3056	3057	3058	3059	3060	3061	3062	3063	3064	3065	3066	3067	3068	3069	3070	3071	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087
9	8140	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103	3104	3105	3106	3107	3108	3109	3110	3111	3112	3113	3114	3115	3116	3117	3118	3119	3120	3121	3122	3123	3124	3125	3126	3127
10	8168	3128	3129	3130	3131	3132	3133	3134	3135	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167
11	8190	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183	3184	3185	3186	3187	3188	3189	3190	3191	3192	3193	3194	3195	3196	3197	3198	3199	3200	3201	3202	3203	3204	3205	3206	3207
12	81B8	3208	3209	3210	3211	3212	3213	3214	3215	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247
13	81E0	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263	3264	3265	3266	3267	3268	3269	3270	3271	3272	3273	3274	3275	3276	3277	3278	3279	3280	3281	3282	3283	3284	3285	3286	3287
14	8208	3288	3289	3290	3291	3292	3293	3294	3295	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327
15	8230	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343	3344	3345	3346	3347	3348	3349	3350	3351	3352	3353	3354	3355	3356	3357	3358	3359	3360	3361	3362	3363	3364	3365	3366	3367
16	8258	3368	3369	3370	3371	3372	3373	3374	3375	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407
17	8280	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423	3424	3425	3426	3427	3428	3429	3430	3431	3432	3433	3434	3435	3436	3437	3438	3439	3440	3441	3442	3443	3444	3445	3446	3447
18	82A8	3448	3449	3450	3451	3452	3453	3454	3455	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487
19	82D0	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503	3504	3505	3506	3507	3508	3509	3510	3511	3512	3513	3514	3515	3516	3517	3518	3519	3520	3521	3522	3523	3524	3525	3526	3527
20	82F8	3528	3529	3530	3531	3532	3533	3534	3535	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567
21	8320	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583	3584	3585	3586	3587	3588	3589	3590	3591	3592	3593	3594	3595	3596	3597	3598	3599	3600	3601	3602	3603	3604	3605	3606	3607
22	8348	3608	3609	3610	3611	3612	3613	3614	3615	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647
23	8370	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663	3664	3665	3666	3667	3668	3669	3670	3671	3672	3673	3674	3675	3676	3677	3678	3679	3680	3681	3682	3683	3684	3685	3686	3687
24	8398	3688	3689	3690	3691	3692	3693	3694	3695	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727
25	83C0	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743	3744	3745	3746	3747	3748	3749	3750	3751	3752	3753	3754	3755	3756	3757	3758	3759	3760	3761	3762	3763	3764	3765	3766	3767

33767

VIC 20 Screen Map (without expansion memory)

7680

1	1E00	7680	7681	7682	7683	7684	7685	7686	7687	7688	7689	7690	7691	7692	7693	7694	7695	7696	7697	7698	7699	7700	7701
2	1E16	7702	7703	7704	7705	7706	7707	7708	7709	7710	7711	7712	7713	7714	7715	7716	7717	7718	7719	7720	7721	7722	7723
3	1E2C	7724	7725	7726	7727	7728	7729	7730	7731	7732	7733	7734	7735	7736	7737	7738	7739	7740	7741	7742	7743	7744	7745
4	1E42	7746	7747	7748	7749	7750	7751	7752	7753	7754	7755	7756	7757	7758	7759	7760	7761	7762	7763	7764	7765	7766	7767
5	1E58	7768	7769	7770	7771	7772	7773	7774	7775	7776	7777	7778	7779	7780	7781	7782	7783	7784	7785	7786	7787	7788	7789
6	1E6E	7790	7791	7792	7793	7794	7795	7796	7797	7798	7799	7800	7801	7802	7803	7804	7805	7806	7807	7808	7809	7810	7811
7	1E84	7812	7813	7814	7815	7816	7817	7818	7819	7820	7821	7822	7823	7824	7825	7826	7827	7828	7829	7830	7831	7832	7833
8	1E9A	7834	7835	7836	7837	7838	7839	7840	7841	7842	7843	7844	7845	7846	7847	7848	7849	7850	7851	7852	7853	7854	7855
9	1E60	7856	7857	7858	7859	7860	7861	7862	7863	7864	7865	7866	7867	7868	7869	7870	7871	7872	7873	7874	7875	7876	7877
10	1EC6	7878	7879	7880	7881	7882	7883	7884	7885	7886	7887	7888	7889	7890	7891	7892	7893	7894	7895	7896	7897	7898	7899
11	1EDC	7900	7901	7902	7903	7904	7905	7906	7907	7908	7909	7910	7911	7912	7913	7914	7915	7916	7917	7918	7919	7920	7921
12	1EF2	7922	7923	7924	7925	7926	7927	7928	7929	7930	7931	7932	7933	7934	7935	7936	7937	7938	7939	7940	7941	7942	7943
13	1F08	7944	7945	7946	7947	7948	7949	7950	7951	7952	7953	7954	7955	7956	7957	7958	7959	7960	7961	7962	7963	7964	7965
14	1F1E	7966	7967	7968	7969	7970	7971	7972	7973	7974	7975	7976	7977	7978	7979	7980	7981	7982	7983	7984	7985	7986	7987
15	1F34	7988	7989	7990	7991	7992	7993	7994	7995	7996	7997	7998	7999	8000	8001	8002	8003	8004	8005	8006	8007	8008	8009
16	1F4A	8010	8011	8012	8013	8014	8015	8016	8017	8018	8019	8020	8021	8022	8023	8024	8025	8026	8027	8028	8029	8030	8031
17	1F60	8032	8033	8034	8035	8036	8037	8038	8039	8040	8041	8042	8043	8044	8045	8046	8047	8048	8049	8050	8051	8052	8053
18	1F76	8054	8055	8056	8057	8058	8059	8060	8061	8062	8063	8064	8065	8066	8067	8068	8069	8070	8071	8072	8073	8074	8075
19	1F8C	8076	8077	8078	8079	8080	8081	8082	8083	8084	8085	8086	8087	8088	8089	8090	8091	8092	8093	8094	8095	8096	8097
20	1FA2	8098	8099	8100	8101	8102	8103	8104	8105	8106	8107	8108	8109	8110	8111	8112	8113	8114	8115	8116	8117	8118	8119
21	1F68	8120	8121	8122	8123	8124	8125	8126	8127	8128	8129	8130	8131	8132	8133	8134	8135	8136	8137	8138	8139	8140	8141
22	1FCE	8142	8143	8144	8145	8146	8147	8148	8149	8150	8151	8152	8153	8154	8155	8156	8157	8158	8159	8160	8161	8162	8163
23	1FE4	8164	8165	8166	8167	8168	8169	8170	8171	8172	8173	8174	8175	8176	8177	8178	8179	8180	8181	8182	8183	8184	8185

8185

VIC 20 Colour Table Map (without expansion memory)

38400

1	9600	8400	8401	8402	8403	8404	8405	8406	8407	8408	8409	8410	8411	8412	8413	8414	8415	8416	8417	8418	8419	8420	8421
2	9616	8422	8423	8424	8425	8426	8427	8428	8429	8430	8431	8432	8433	8434	8435	8436	8437	8438	8439	8440	8441	8442	8443
3	962C	8444	8445	8446	8447	8448	8449	8450	8451	8452	8453	8454	8455	8456	8457	8458	8459	8460	8461	8462	8463	8464	8465
4	9642	8466	8467	8468	8469	8470	8471	8472	8473	8474	8475	8476	8477	8478	8479	8480	8481	8482	8483	8484	8485	8486	8487
5	9658	8488	8489	8490	8491	8492	8493	8494	8495	8496	8497	8498	8499	8500	8501	8502	8503	8504	8505	8506	8507	8508	8509
6	966E	8510	8511	8512	8513	8514	8515	8516	8517	8518	8519	8520	8521	8522	8523	8524	8525	8526	8527	8528	8529	8530	8531
7	9684	8532	8533	8534	8535	8536	8537	8538	8539	8540	8541	8542	8543	8544	8545	8546	8547	8548	8549	8550	8551	8552	8553
8	969A	8554	8555	8556	8557	8558	8559	8560	8561	8562	8563	8564	8565	8566	8567	8568	8569	8570	8571	8572	8573	8574	8575
9	9660	8576	8577	8578	8579	8580	8581	8582	8583	8584	8585	8586	8587	8588	8589	8590	8591	8592	8593	8594	8595	8596	8597
10	96C6	8598	8599	8600	8601	8602	8603	8604	8605	8606	8607	8608	8609	8610	8611	8612	8613	8614	8615	8616	8617	8618	8619
11	96DC	8620	8621	8622	8623	8624	8625	8626	8627	8628	8629	8630	8631	8632	8633	8634	8635	8636	8637	8638	8639	8640	8641
12	96F2	8642	8643	8644	8645	8646	8647	8648	8649	8650	8651	8652	8653	8654	8655	8656	8657	8658	8659	8660	8661	8662	8663
13	9708	8664	8665	8666	8667	8668	8669	8670	8671	8672	8673	8674	8675	8676	8677	8678	8679	8680	8681	8682	8683	8684	8685
14	971E	8686	8687	8688	8689	8690	8691	8692	8693	8694	8695	8696	8697	8698	8699	8700	8701	8702	8703	8704	8705	8706	8707
15	9734	8708	8709	8710	8711	8712	8713	8714	8715	8716	8717	8718	8719	8720	8721	8722	8723	8724	8725	8726	8727	8728	8729
16	974A	8730	8731	8732	8733	8734	8735	8736	8737	8738	8739	8740	8741	8742	8743	8744	8745	8746	8747	8748	8749	8750	8751
17	9760	8752	8753	8754	8755	8756	8757	8758	8759	8760	8761	8762	8763	8764	8765	8766	8767	8768	8769	8770	8771	8772	8773
18	9776	8774	8775	8776	8777	8778	8779	8780	8781	8782	8783	8784	8785	8786	8787	8788	8789	8790	8791	8792	8793	8794	8795
19	978C	8796	8797	8798	8799	8800	8801	8802	8803	8804	8805	8806	8807	8808	8809	8810	8811	8812	8813	8814	8815	8816	8817
20	97A2	8818	8819	8820	8821	8822	8823	8824	8825	8826	8827	8828	8829	8830	8831	8832	8833	8834	8835	8836	8837	8838	8839
21	9768	8840	8841	8842	8843	8844	8845	8846	8847	8848	8849	8850	8851	8852	8853	8854	8855	8856	8857	8858	8859	8860	8861
22	97CE	8862	8863	8864	8865	8866	8867	8868	8869	8870	8871	8872	8873	8874	8875	8876	8877	8878	8879	8880	8881	8882	8883
23	97E4	8884	8885	8886	8887	8888	8889	8890	8891	8892	8893	8894	8895	8896	8897	8898	8899	8900	8901	8902	8903	8904	8905

38905

VIC 20 Screen Map (with expansion memory at $2000)

4096

1	1000	4096	4097	4098	4099	4100	4101	4102	4103	4104	4105	4106	4107	4108	4109	4110	4111	4112	4113	4114	4115	4116	4117
2	1016	4118	4119	4120	4121	4122	4123	4124	4125	4126	4127	4128	4129	4130	4131	4132	4133	4134	4135	4136	4137	4138	4139
3	102C	4140	4141	4142	4143	4144	4145	4146	4147	4148	4149	4150	4151	4152	4153	4154	4155	4156	4157	4158	4159	4160	4161
4	1042	4162	4163	4164	4165	4166	4167	4168	4169	4170	4171	4172	4173	4174	4175	4176	4177	4178	4179	4180	4181	4182	4183
5	1058	4184	4185	4186	4187	4188	4189	4190	4191	4192	4193	4194	4195	4196	4197	4198	4199	4200	4201	4202	4203	4204	4205
6	106E	4206	4207	4208	4209	4210	4211	4212	4213	4214	4215	4216	4217	4218	4219	4220	4221	4222	4223	4224	4225	4226	4227
7	1084	4228	4229	4230	4231	4232	4233	4234	4235	4236	4237	4238	4239	4240	4241	4242	4243	4244	4245	4246	4247	4248	4249
8	109A	4250	4251	4252	4253	4254	4255	4256	4257	4258	4259	4260	4261	4262	4263	4264	4265	4266	4267	4268	4269	4270	4271
9	1060	4272	4273	4274	4275	4276	4277	4278	4279	4280	4281	4282	4283	4284	4285	4286	4287	4288	4289	4290	4291	4292	4293
10	10C6	4294	4295	4296	4297	4298	4299	4300	4301	4302	4303	4304	4305	4306	4307	4308	4309	4310	4311	4312	4313	4314	4315
11	10DC	4316	4317	4318	4319	4320	4321	4322	4323	4324	4325	4326	4327	4328	4329	4330	4331	4332	4333	4334	4335	4336	4337
12	10F2	4338	4339	4340	4341	4342	4343	4344	4345	4346	4347	4348	4349	4350	4351	4352	4353	4354	4355	4356	4357	4358	4359
13	1108	4360	4361	4362	4363	4364	4365	4366	4367	4368	4369	4370	4371	4372	4373	4374	4375	4376	4377	4378	4379	4380	4381
14	111E	4382	4383	4384	4385	4386	4387	4388	4389	4390	4391	4392	4393	4394	4395	4396	4397	4398	4399	4400	4401	4402	4403
15	1134	4404	4405	4406	4407	4408	4409	4410	4411	4412	4413	4414	4415	4416	4417	4418	4419	4420	4421	4422	4423	4424	4425
16	114A	4426	4427	4428	4429	4430	4431	4432	4433	4434	4435	4436	4437	4438	4439	4440	4441	4442	4443	4444	4445	4446	4447
17	1160	4448	4449	4450	4451	4452	4453	4454	4455	4456	4457	4458	4459	4460	4461	4462	4463	4464	4465	4466	4467	4468	4469
18	1176	4470	4471	4472	4473	4474	4475	4476	4477	4478	4479	4480	4481	4482	4483	4484	4485	4486	4487	4488	4489	4490	4491
19	118C	4492	4493	4494	4495	4496	4497	4498	4499	4500	4501	4502	4503	4504	4505	4506	4507	4508	4509	4510	4511	4512	4513
20	11A2	4514	4515	4516	4517	4518	4519	4520	4521	4522	4523	4524	4525	4526	4527	4528	4529	4530	4531	4532	4533	4534	4535
21	1168	4536	4537	4538	4539	4540	4541	4542	4543	4544	4545	4546	4547	4548	4549	4550	4551	4552	4553	4554	4555	4556	4557
22	11CE	4558	4559	4560	4561	4562	4563	4564	4565	4566	4567	4568	4569	4570	4571	4572	4573	4574	4575	4576	4577	4578	4579
23	11E4	4580	4581	4582	4583	4584	4585	4586	4587	4588	4589	4590	4591	4592	4593	4594	4595	4596	4597	4598	4599	4600	4601

4601

VIC 20 Colour Table Map (with expansion memory)

37888

1	9400	7888	7889	7890	7891	7892	7893	7894	7895	7896	7897	7898	7899	7900	7901	7902	7903	7904	7905	7906	7907	7908	7909
2	9416	7910	7911	7912	7913	7914	7915	7916	7917	7918	7919	7920	7921	7922	7923	7924	7925	7926	7927	7928	7929	7930	7931
3	942C	7932	7933	7934	7935	7936	7937	7938	7939	7940	7941	7942	7943	7944	7945	7946	7947	7948	7949	7950	7951	7952	7953
4	9442	7954	7955	7956	7957	7958	7959	7960	7961	7962	7963	7964	7965	7966	7967	7968	7969	7970	7971	7972	7973	7974	7975
5	9458	7976	7977	7978	7979	7980	7981	7982	7983	7984	7985	7986	7987	7988	7989	7990	7991	7992	7993	7994	7995	7996	7997
6	946E	7998	7999	8000	8001	8002	8003	8004	8005	8006	8007	8008	8009	8010	8011	8012	8013	8014	8015	8016	8017	8018	8019
7	9484	8020	8021	8022	8023	8024	8025	8026	8027	8028	8029	8030	8031	8032	8033	8034	8035	8036	8037	8038	8039	8040	8041
8	949A	8042	8043	8044	8045	8046	8047	8048	8049	8050	8051	8052	8053	8054	8055	8056	8057	8058	8059	8060	8061	8062	8063
9	9460	8064	8065	8066	8067	8068	8069	8070	8071	8072	8073	8074	8075	8076	8077	8078	8079	8080	8081	8082	8083	8084	8085
10	94C6	8086	8087	8088	8089	8090	8091	8092	8093	8094	8095	8096	8097	8098	8099	8100	8101	8102	8103	8104	8105	8106	8107
11	94DC	8108	8109	8110	8111	8112	8113	8114	8115	8116	8117	8118	8119	8120	8121	8122	8123	8124	8125	8126	8127	8128	8129
12	94F2	8130	8131	8132	8133	8134	8135	8136	8137	8138	8139	8140	8141	8142	8143	8144	8145	8146	8147	8148	8149	8150	8151
13	9508	8152	8153	8154	8155	8156	8157	8158	8159	8160	8161	8162	8163	8164	8165	8166	8167	8168	8169	8170	8171	8172	8173
14	951E	8174	8175	8176	8177	8178	8179	8180	8181	8182	8183	8184	8185	8186	8187	8188	8189	8190	8191	8192	8193	8194	8195
15	9534	8196	8197	8198	8199	8200	8201	8202	8203	8204	8205	8206	8207	8208	8209	8210	8211	8212	8213	8214	8215	8216	8217
16	954A	8218	8219	8220	8221	8222	8223	8224	8225	8226	8227	8228	8229	8230	8231	8232	8233	8234	8235	8236	6237	8238	8239
17	9560	8240	8241	8242	8243	8244	8245	8246	8247	8248	8249	8250	8251	8252	8253	8254	8255	8256	8257	8258	8259	8260	8261
18	9576	8262	8263	8264	8265	8266	8267	8268	8269	8270	8271	8272	8273	8274	8275	8276	8277	8278	8279	8280	8281	8282	8263
19	958C	8284	8285	8286	8287	8288	8289	8290	8291	8292	8293	8294	8295	8296	8297	8298	8299	8300	8301	8302	8303	8304	8305
20	95A2	8306	8307	8308	8309	8310	8311	8312	8313	8314	8315	8316	8317	8318	8319	8320	8321	8322	8323	8324	8325	8326	8327
21	9568	8328	8329	8330	8331	8332	8333	8334	8335	8336	8337	8338	8339	8340	8341	8342	8343	8344	8345	8346	8347	8348	8349
22	95CE	8350	8351	8352	8353	8354	8355	8356	8357	8358	8359	8360	8361	8362	8363	8364	8365	8366	8367	8368	8369	8370	8371
23	95E4	8372	8373	8374	8375	8376	8377	8378	8379	8380	8381	8382	8383	8384	8385	8386	8387	8388	8389	8390	8391	8392	8393

38393

B Series Screen Map

53248

1	D000	53248	53249	53250	53251	53252	53253	53254	53255	53256	53257	53258	53259	53260	53261	53262	53263	53264	53265	53266	53267
2	D050	53328	53329	53330	53331	53332	53333	53334	53335	53336	53337	53338	53339	53340	53341	53342	53343	53344	53345	53346	53347
3	D0A0	53408	53409	53410	53411	53412	53413	53414	53415	53416	53417	53418	53419	53420	53421	53422	53423	53424	53425	53426	53427
4	D0F0	53488	53489	53490	53491	53492	53493	53494	53495	53496	53497	53498	53499	53500	53501	53502	53503	53504	53505	53506	53507
5	D140	53568	53569	53570	53571	53572	53573	53574	53575	53576	53577	53578	53579	53580	53581	53582	53583	53584	53585	53586	53587
6	D190	53648	53649	53650	53651	53652	53653	53654	53655	53656	53657	53658	53659	53660	53661	53662	53663	53664	53665	53666	53667
7	D1E0	53728	53729	53730	53731	53732	53733	53734	53735	53736	53737	53738	53739	53740	53741	53742	53743	53744	53745	53746	53747
8	D230	53808	53809	53810	53811	53812	53813	53814	53815	53816	53817	53818	53819	53820	53821	53822	53823	53824	53825	53826	53827
9	D280	53888	53889	53890	53891	53892	53893	53894	53895	53896	53897	53898	53899	53900	53901	53902	53903	53904	53905	53906	53907
10	D2D0	53968	53969	53970	53971	53972	53973	53974	53975	53976	53977	53978	53979	53980	53981	53982	53983	53984	53985	53986	53987
11	D320	54048	54049	54050	54051	54052	54053	54054	54055	54056	54057	54058	54059	54060	54061	54062	54063	54064	54065	54066	54067
12	D370	54128	54129	54130	54131	54132	54133	54134	54135	54136	54137	54138	54139	54140	54141	54142	54143	54144	54145	54146	54147
13	D3C0	54208	54209	54210	54211	54212	54213	54214	54215	54216	54217	54218	54219	54220	54221	54222	54223	54224	54225	54226	54227
14	D410	54288	54289	54290	54291	54292	54293	54294	54295	54296	54297	54298	54299	54300	54301	54302	54303	54304	54305	54306	54307
15	D460	54368	54369	54370	54371	54372	54373	54374	54375	54376	54377	54378	54379	54380	54381	54382	54383	54384	54385	54386	54387
16	D4B0	54448	54449	54450	54451	54452	54453	54454	54455	54456	54457	54458	54459	54460	54461	54462	54463	54464	54465	54466	54467
17	D500	54528	54529	54530	54531	54532	54533	54534	54535	54536	54537	54538	54539	54540	54541	54542	54543	54544	54545	54546	54547
18	D550	54608	54609	54610	54611	54612	54613	54614	54615	54616	54617	54618	54619	54620	54621	54622	54623	54624	54625	54626	54627
19	D5A0	54688	54689	54690	54691	54692	54693	54694	54695	54696	54697	54698	54699	54700	54701	54702	54703	54704	54705	54706	54707
20	D5F0	54768	54769	54770	54771	54772	54773	54774	54775	54776	54777	54778	54779	54780	54781	54782	54783	54784	54785	54786	54787
21	D640	54848	54849	54850	54851	54852	54853	54854	54855	54856	54857	54858	54859	54860	54861	54862	54863	54864	54865	54866	54867
22	D690	54928	54929	54930	54931	54932	54933	54934	54935	54936	54937	54938	54939	54940	54941	54942	54943	54944	54945	54946	54947
23	D6E0	55008	55009	55010	55011	55012	55013	55014	55015	55016	55017	55018	55019	55020	55021	55022	55023	55024	55025	55026	55027
24	D730	55088	55089	55090	55091	55092	55093	55094	55095	55096	55097	55098	55099	55100	55101	55102	55103	55104	55105	55106	55107
25	D780	55168	55169	55170	55171	55172	55173	55174	55175	55176	55177	55178	55179	55180	55181	55182	55183	55184	55185	55186	55187

1	D000	53268	53269	53270	53271	53272	53273	53274	53275	53276	53277	53278	53279	53280	53281	53282	53283	53284	53285	53286	53287
2	D050	53348	53349	53350	53351	53352	53353	53354	53355	53356	53357	53358	53359	53360	53361	53362	53363	53364	53365	53366	53367
3	D0A0	53428	53429	53430	53431	53432	53433	53434	53435	53436	53437	53438	53439	53440	53441	53442	53443	53444	53445	53446	53447
4	D0F0	53508	53509	53510	53511	53512	53513	53514	53515	53516	53517	53518	53519	53520	53521	53522	53523	53524	53525	53526	53527
5	D140	53588	53589	53590	53591	53592	53593	53594	53595	53596	53597	53598	53599	53600	53601	53602	53603	53604	53605	53606	53607
6	D190	53668	53669	53670	53671	53672	53673	53674	53675	53676	53677	53678	53679	53680	53681	53682	53683	53684	53685	53686	53687
7	D1E0	53748	53749	53750	53751	53752	53753	53754	53755	53756	53757	53758	53759	53760	53761	53762	53763	53764	53765	53766	53767
8	D230	53828	53829	53830	53831	53832	53833	53834	53835	53836	53837	53838	53839	53840	53841	53842	53843	53844	53845	53846	53847
9	D280	53908	53909	53910	53911	53912	53913	53914	53915	53916	53917	53918	53919	53920	53921	53922	53923	53924	53925	53926	53927
10	D2D0	53988	53989	53990	53991	53992	53993	53994	53995	53996	53997	53998	53999	54000	54001	54002	54003	54004	54005	54006	54007
11	D320	54068	54069	54070	54071	54072	54073	54074	54075	54076	54077	54078	54079	54080	54081	54082	54083	54084	54085	54086	54087
12	D370	54148	54149	54150	54151	54152	54153	54154	54155	54156	54157	54158	54159	54160	54161	54162	54163	54164	54165	54166	54167
13	D3C0	54228	54229	54230	54231	54232	54233	54234	54235	54236	54237	54238	54239	54240	54241	54242	54243	54244	54245	54246	54247
14	D410	54308	54309	54310	54311	54312	54313	54314	54315	54316	54317	54318	54319	54320	54321	54322	54323	54324	54325	54326	54327
15	D460	54388	54389	54390	54391	54392	54393	54394	54395	54396	54397	54398	54399	54400	54401	54402	54403	54404	54405	54406	54407
16	D4B0	54468	54469	54470	54471	54472	54473	54474	54475	54476	54477	54478	54479	54480	54481	54482	54483	54484	54485	54486	54487
17	D500	54548	54549	54550	54551	54552	54553	54554	54555	54556	54557	54558	54559	54560	54561	54562	54563	54564	54565	54566	54567
18	D550	54628	54629	54630	54631	54632	54633	54634	54635	54636	54637	54638	54639	54640	54641	54642	54643	54644	54645	54646	54647
19	D5A0	54708	54709	54710	54711	54712	54713	54714	54715	54716	54717	54718	54719	54720	54721	54722	54723	54724	54725	54726	54727
20	D5F0	54788	54789	54790	54791	54792	54793	54794	54795	54796	54797	54798	54799	54800	54801	54802	54803	54804	54805	54806	54807
21	D640	54868	54869	54870	54871	54872	54873	54874	54875	54876	54877	54878	54879	54880	54881	54882	54883	54884	54885	54886	54887
22	D690	54948	54949	54950	54951	54952	54953	54954	54955	54956	54957	54958	54959	54960	54961	54962	54963	54964	54965	54966	54967
23	D6E0	55028	55029	55030	55031	55032	55033	55034	55035	55036	55037	55038	55039	55040	55041	55042	55043	55044	55045	55046	55047
24	D730	55108	55109	55110	55111	55112	55113	55114	55115	55116	55117	55118	55119	55120	55121	55122	55123	55124	55125	55126	55127
25	D780	55188	55189	55190	55191	55192	55193	55194	55195	55196	55197	55198	55199	55200	55201	55202	55203	55204	55205	55206	55207

1	D000	53288	53289	53290	53291	53292	53293	53294	53295	53296	53297	53298	53299	53300	53301	53302	53303	53304	53305	53306	53307
2	D050	53368	53369	53370	53371	53372	53373	53374	53375	53376	53377	53378	53379	53380	53381	53382	53383	53384	53385	53386	53387
3	D0A0	53448	53449	53450	53451	53452	53453	53454	53455	53456	53457	53458	53459	53460	53461	53462	53463	53464	53465	53466	53467
4	D0F0	53528	53529	53530	53531	53532	53533	53534	53535	53536	53537	53538	53539	53540	53541	53542	53543	53544	53545	53546	53547
5	D140	53608	53609	53610	53611	53612	53613	53614	53615	53616	53617	53618	53619	53620	53621	53622	53623	53624	53625	53626	53627
6	D190	53688	53689	53690	53691	53692	53693	53694	53695	53696	53697	53698	53699	53700	53701	53702	53703	53704	53705	53706	53707
7	D1E0	53768	53769	53770	53771	53772	53773	53774	53775	53776	53777	53778	53779	53780	53781	53782	53783	53784	53785	53786	53787
8	D230	53848	53849	53850	53851	53852	53853	53854	53855	53856	53857	53858	53859	53860	53861	53862	53863	53864	53865	53866	53867
9	D280	53928	53929	53930	53931	53932	53933	53934	53935	53936	53937	53938	53939	53940	53941	53942	53943	53944	53945	53946	53947
10	D2D0	54008	54009	54010	54011	54012	54013	54014	54015	54016	54017	54018	54019	54020	54021	54022	54023	54024	54025	54026	54027
11	D320	54088	54089	54090	54091	54092	54093	54094	54095	54096	54097	54098	54099	54100	54101	54102	54103	54104	54105	54106	54107
12	D370	54168	54169	54170	54171	54172	54173	54174	54175	54176	54177	54178	54179	54180	54181	54182	54183	54184	54185	54186	54187
13	D3C0	54248	54249	54250	54251	54252	54253	54254	54255	54256	54257	54258	54259	54260	54261	54262	54263	54264	54265	54266	54267
14	D410	54328	54329	54330	54331	54332	54333	54334	54335	54336	54337	54338	54339	54340	54341	54342	54343	54344	54345	54346	54347
15	D460	54408	54409	54410	54411	54412	54413	54414	54415	54416	54417	54418	54419	54420	54421	54422	54423	54424	54425	54426	54427
16	D4B0	54488	54489	54490	54491	54492	54493	54494	54495	54496	54497	54498	54499	54500	54501	54502	54503	54504	54505	54506	54507
17	D500	54568	54569	54570	54571	54572	54573	54574	54575	54576	54577	54578	54579	54580	54581	54582	54583	54584	54585	54586	54587
18	D550	54648	54649	54650	54651	54652	54653	54654	54655	54656	54657	54658	54659	54660	54661	54662	54663	54664	54665	54666	54667
19	D5A0	54728	54729	54730	54731	54732	54733	54734	54735	54736	54737	54738	54739	54740	54741	54742	54743	54744	54745	54746	54747
20	D5F0	54808	54809	54810	54811	54812	54813	54814	54815	54816	54817	54818	54819	54820	54821	54822	54823	54824	54825	54826	54827
21	D640	54888	54889	54890	54891	54892	54893	54894	54895	54896	54897	54898	54899	54900	54901	54902	54903	54904	54905	54906	54907
22	D690	54968	54969	54970	54971	54972	54973	54974	54975	54976	54977	54978	54979	54980	54981	54982	54983	54984	54985	54986	54987
23	D6E0	55048	55049	55050	55051	55052	55053	55054	55055	55056	55057	55058	55059	55060	55061	55062	55063	55064	55065	55066	55067
24	D730	55128	55129	55130	55131	55132	55133	55134	55135	55136	55137	55138	55139	55140	55141	55142	55143	55144	55145	55146	55147
25	D780	55208	55209	55210	55211	55212	55213	55214	55215	55216	55217	55218	55219	55220	55221	55222	55223	55224	55225	55226	55227

1	D000	53308	53309	53310	53311	53312	53313	53314	53315	53316	53317	53318	53319	53320	53321	53322	53323	53324	53325	53326	53327
2	D050	53388	53389	53390	53391	53392	53393	53394	53395	53396	53397	53398	53399	53400	53401	53402	53403	53404	53405	53406	53407
3	D0A0	53468	53469	53470	53471	53472	53473	53474	53475	53476	53477	53478	53479	53480	53481	53482	53483	53484	53485	53486	53487
4	D0F0	53548	53549	53550	53551	53552	53553	53554	53555	53556	53557	53558	53559	53560	53561	53562	53563	53564	53565	53566	53567
5	D140	53628	53629	53630	53631	53632	53633	53634	53635	53636	53637	53638	53639	53640	53641	53642	53643	53644	53645	53646	53647
6	D190	53708	53709	53710	53711	53712	53713	53714	53715	53716	53717	53718	53719	53720	53721	53722	53723	53724	53725	53726	53727
7	D1E0	53788	53789	53790	53791	53792	53793	53794	53795	53796	53797	53798	53799	53800	53801	53802	53803	53804	53805	53806	53807
8	D230	53868	53869	53870	53871	53872	53873	53874	53875	53876	53877	53878	53879	53880	53881	53882	53883	53884	53885	53886	53887
9	D280	53948	53949	53950	53951	53952	53953	53954	53955	53956	53957	53958	53959	53960	53961	53962	53963	53964	53965	53966	53967
10	D2D0	54028	54029	54030	54031	54032	54033	54034	54035	54036	54037	54038	54039	54040	54041	54042	54043	54044	54045	54046	54047
11	D320	54108	54109	54110	54111	54112	54113	54114	54115	54116	54117	54118	54119	54120	54121	54122	54123	54124	54125	54126	54127
12	D370	54188	54189	54190	54191	54192	54193	54194	54195	54196	54197	54198	54199	54200	54201	54202	54203	54204	54205	54206	54207
13	D3C0	54268	54269	54270	54271	54272	54273	54274	54275	54276	54277	54278	54279	54280	54281	54282	54283	54284	54285	54286	54287
14	D410	54348	54349	54350	54351	54352	54353	54354	54355	54356	54357	54358	54359	54360	54361	54362	54363	54364	54365	54366	54367
15	D460	54428	54429	54430	54431	54432	54433	54434	54435	54436	54437	54438	54439	54440	54441	54442	54443	54444	54445	54446	54447
16	D4B0	54508	54509	54510	54511	54512	54513	54514	54515	54516	54517	54518	54519	54520	54521	54522	54523	54524	54525	54526	54527
17	D500	54588	54589	54590	54591	54592	54593	54594	54595	54596	54597	54598	54599	54600	54601	54602	54603	54604	54605	54606	54607
18	D550	54668	54669	54670	54671	54672	54673	54674	54675	54676	54677	54678	54679	54680	54681	54682	54683	54684	54685	54686	54687
19	D5A0	54748	54749	54750	54751	54752	54753	54754	54755	54756	54757	54758	54759	54760	54761	54762	54763	54764	54765	54766	54767
20	D5F0	54828	54829	54830	54831	54832	54833	54834	54835	54836	54837	54838	54839	54840	54841	54842	54843	54844	54845	54846	54847
21	D640	54908	54909	54910	54911	54912	54913	54914	54915	54916	54917	54918	54919	54920	54921	54922	54923	54924	54925	54926	54927
22	D690	54988	54989	54990	54991	54992	54993	54994	54995	54996	54997	54998	54999	55000	55001	55002	55003	55004	55005	55006	55007
23	D6E0	55068	55069	55070	55071	55072	55073	55074	55075	55076	55077	55078	55079	55080	55081	55082	55083	55084	55085	55086	55087
24	D730	55148	55149	55150	55151	55152	55153	55154	55155	55156	55157	55158	55159	55160	55161	55162	55163	55164	55165	55166	55167
25	D780	55228	55229	55230	55231	55232	55233	55234	55235	55236	55237	55238	55239	55240	55241	55242	55243	55244	55245	55246	55247

55247

+4 / C16 Screen Map

3072

1	0C00	3072	3073	3074	3075	3076	3077	3078	3079	3080	3081	3082	3083	3084	3085	3086	3087	3088	3089	3090	3091	3092	3093	3094	3095	3096	3097	3098	3099	3100	3101	3102	3103	3104	3105	3106	3107	3108	3109	3110	3111
2	0C28	3112	3113	3114	3115	3116	3117	3118	3119	3120	3121	3122	3123	3124	3125	3126	3127	3128	3129	3130	3131	3132	3133	3134	3135	3136	3137	3138	3139	3140	3141	3142	3143	3144	3145	3146	3147	3148	3149	3150	3151
3	0C50	3152	3153	3154	3155	3156	3157	3158	3159	3160	3161	3162	3163	3164	3165	3166	3167	3168	3169	3170	3171	3172	3173	3174	3175	3176	3177	3178	3179	3180	3181	3182	3183	3184	3185	3186	3187	3188	3189	3190	3191
4	0C78	3192	3193	3194	3195	3196	3197	3198	3199	3200	3201	3202	3203	3204	3205	3206	3207	3208	3209	3210	3211	3212	3213	3214	3215	3216	3217	3218	3219	3220	3221	3222	3223	3224	3225	3226	3227	3228	3229	3230	3231
5	0CA0	3232	3233	3234	3235	3236	3237	3238	3239	3240	3241	3242	3243	3244	3245	3246	3247	3248	3249	3250	3251	3252	3253	3254	3255	3256	3257	3258	3259	3260	3261	3262	3263	3264	3265	3266	3267	3268	3269	3270	3271
6	0CC8	3272	3273	3274	3275	3276	3277	3278	3279	3280	3281	3282	3283	3284	3285	3286	3287	3288	3289	3290	3291	3292	3293	3294	3295	3296	3297	3298	3299	3300	3301	3302	3303	3304	3305	3306	3307	3308	3309	3310	3311
7	0CF0	3312	3313	3314	3315	3316	3317	3318	3319	3320	3321	3322	3323	3324	3325	3326	3327	3328	3329	3330	3331	3332	3333	3334	3335	3336	3337	3338	3339	3340	3341	3342	3343	3344	3345	3346	3347	3348	3349	3350	3351
8	0D18	3352	3353	3354	3355	3356	3357	3358	3359	3360	3361	3362	3363	3364	3365	3366	3367	3368	3369	3370	3371	3372	3373	3374	3375	3376	3377	3378	3379	3380	3381	3382	3383	3384	3385	3386	3387	3388	3389	3390	3391
9	0D40	3392	3393	3394	3395	3396	3397	3398	3399	3400	3401	3402	3403	3404	3405	3406	3407	3408	3409	3410	3411	3412	3413	3414	3415	3416	3417	3418	3419	3420	3421	3422	3423	3424	3425	3426	3427	3428	3429	3430	3431
10	0D68	3432	3433	3434	3435	3436	3437	3438	3439	3440	3441	3442	3443	3444	3445	3446	3447	3448	3449	3450	3451	3452	3453	3454	3455	3456	3457	3458	3459	3460	3461	3462	3463	3464	3465	3466	3467	3468	3469	3470	3471
11	0D90	3472	3473	3474	3475	3476	3477	3478	3479	3480	3481	3482	3483	3484	3485	3486	3487	3488	3489	3490	3491	3492	3493	3494	3495	3496	3497	3498	3499	3500	3501	3502	3503	3504	3505	3506	3507	3508	3509	3510	3511
12	0DB8	3512	3513	3514	3515	3516	3517	3518	3519	3520	3521	3522	3523	3524	3525	3526	3527	3528	3529	3530	3531	3532	3533	3534	3535	3536	3537	3538	3539	3540	3541	3542	3543	3544	3545	3546	3547	3548	3549	3550	3551
13	0DE0	3552	3553	3554	3555	3556	3557	3558	3559	3560	3561	3562	3563	3564	3565	3566	3567	3568	3569	3570	3571	3572	3573	3574	3575	3576	3577	3578	3579	3580	3581	3582	3583	3584	3585	3586	3587	3588	3589	3590	3591
14	0E08	3592	3593	3594	3595	3596	3597	3598	3599	3600	3601	3602	3603	3604	3605	3606	3607	3608	3609	3610	3611	3612	3613	3614	3615	3616	3617	3618	3619	3620	3621	3622	3623	3624	3625	3626	3627	3628	3629	3630	3631
15	0E30	3632	3633	3634	3635	3636	3637	3638	3639	3640	3641	3642	3643	3644	3645	3646	3647	3648	3649	3650	3651	3652	3653	3654	3655	3656	3657	3658	3659	3660	3661	3662	3663	3664	3665	3666	3667	3668	3669	3670	3671
16	0E58	3672	3673	3674	3675	3676	3677	3678	3679	3680	3681	3682	3683	3684	3685	3686	3687	3688	3689	3690	3691	3692	3693	3694	3695	3696	3697	3698	3699	3700	3701	3702	3703	3704	3705	3706	3707	3708	3709	3710	3711
17	0E80	3712	3713	3714	3715	3716	3717	3718	3719	3720	3721	3722	3723	3724	3725	3726	3727	3728	3729	3730	3731	3732	3733	3734	3735	3736	3737	3738	3739	3740	3741	3742	3743	3744	3745	3746	3747	3748	3749	3750	3751
18	0EA8	3752	3753	3754	3755	3756	3757	3758	3759	3760	3761	3762	3763	3764	3765	3766	3767	3768	3769	3770	3771	3772	3773	3774	3775	3776	3777	3778	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
19	0ED0	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3802	3803	3804	3805	3806	3807	3808	3809	3810	3811	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821	3822	3823	3824	3825	3826	3827	3828	3829	3830	3831
20	0EF8	3832	3833	3834	3835	3836	3837	3838	3839	3840	3841	3842	3843	3844	3845	3846	3847	3848	3849	3850	3851	3852	3853	3854	3855	3856	3857	3858	3859	3860	3861	3862	3863	3864	3865	3866	3867	3868	3869	3870	3871
21	0F20	3872	3873	3874	3875	3876	3877	3878	3879	3880	3881	3882	3883	3884	3885	3886	3887	3888	3889	3890	3891	3892	3893	3894	3895	3896	3897	3898	3899	3900	3901	3902	3903	3904	3905	3906	3907	3908	3909	3910	3911
22	0F48	3912	3913	3914	3915	3916	3917	3918	3919	3920	3921	3922	3923	3924	3925	3926	3927	3928	3929	3930	3931	3932	3933	3934	3935	3936	3937	3938	3939	3940	3941	3942	3943	3944	3945	3946	3947	3948	3949	3950	3951
23	0F70	3952	3953	3954	3955	3956	3957	3958	3959	3960	3961	3962	3963	3964	3965	3966	3967	3968	3969	3970	3971	3972	3973	3974	3975	3976	3977	3978	3979	3980	3981	3982	3983	3984	3985	3986	3987	3988	3989	3990	3991
24	0F98	3992	3993	3994	3995	3996	3997	3998	3999	4000	4001	4002	4003	4004	4005	4006	4007	4008	4009	4010	4011	4012	4013	4014	4015	4016	4017	4018	4019	4020	4021	4022	4023	4024	4025	4026	4027	4028	4029	4030	4031
25	0FC0	4032	4033	4034	4035	4036	4037	4038	4039	4040	4041	4042	4043	4044	4045	4046	4047	4048	4049	4050	4051	4052	4053	4054	4055	4056	4057	4058	4059	4060	4061	4062	4063	4064	4065	4066	4067	4068	4069	4070	4071

4071

+4 / C16 Colour Table Map

2087

1	0800	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087
2	0828	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127
3	0850	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159	2160	2161	2162	2163	2164	2165	2166	2167
4	0878	2168	2169	2170	2171	2172	2173	2174	2175	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207
5	08A0	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239	2240	2241	2242	2243	2244	2245	2246	2247
6	08C8	2248	2249	2250	2251	2252	2253	2254	2255	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287
7	08F0	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303	2304	2305	2306	2307	2308	2309	2310	2311	2312	2313	2314	2315	2316	2317	2318	2319	2320	2321	2322	2323	2324	2325	2326	2327
8	0918	2328	2329	2330	2331	2332	2333	2334	2335	2336	2337	2338	2339	2340	2341	2342	2343	2344	2345	2346	2347	2348	2349	2350	2351	2352	2353	2354	2355	2356	2357	2358	2359	2360	2361	2362	2363	2364	2365	2366	2367
9	0940	2368	2369	2370	2371	2372	2373	2374	2375	2376	2377	2378	2379	2380	2381	2382	2383	2384	2385	2386	2387	2388	2389	2390	2391	2392	2393	2394	2395	2396	2397	2398	2399	2400	2401	2402	2403	2404	2405	2406	2407
10	0968	2408	2409	2410	2411	2412	2413	2414	2415	2416	2417	2418	2419	2420	2421	2422	2423	2424	2425	2426	2427	2428	2429	2430	2431	2432	2433	2434	2435	2436	2437	2438	2439	2440	2441	2442	2443	2444	2445	2446	2447
11	0990	2448	2449	2450	2451	2452	2453	2454	2455	2456	2457	2458	2459	2460	2461	2462	2463	2464	2465	2466	2467	2468	2469	2470	2471	2472	2473	2474	2475	2476	2477	2478	2479	2480	2481	2482	2483	2484	2485	2486	2487
12	09B8	2488	2489	2490	2491	2492	2493	2494	2495	2496	2497	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2509	2510	2511	2512	2513	2514	2515	2516	2517	2518	2519	2520	2521	2522	2523	2524	2525	2526	2527
13	09E0	2528	2529	2530	2531	2532	2533	2534	2535	2536	2537	2538	2539	2540	2541	2542	2543	2544	2545	2546	2547	2548	2549	2550	2551	2552	2553	2554	2555	2556	2557	2558	2559	2560	2561	2562	2563	2564	2565	2566	2567
14	0A08	2568	2569	2570	2571	2572	2573	2574	2575	2576	2577	2578	2579	2580	2581	2582	2583	2584	2585	2586	2587	2588	2589	2590	2591	2592	2593	2594	2595	2596	2597	2598	2599	2600	2601	2602	2603	2604	2605	2606	2607
15	0A30	2608	2609	2610	2611	2612	2613	2614	2615	2616	2617	2618	2619	2620	2621	2622	2623	2624	2625	2626	2627	2628	2629	2630	2631	2632	2633	2634	2635	2636	2637	2638	2639	2640	2641	2642	2643	2644	2645	2646	2647
16	0A58	2648	2649	2650	2651	2652	2653	2654	2655	2656	2657	2658	2659	2660	2661	2662	2663	2664	2665	2666	2667	2668	2669	2670	2671	2672	2673	2674	2675	2676	2677	2678	2679	2680	2681	2682	2683	2684	2685	2686	2687
17	0A80	2688	2689	2690	2691	2692	2693	2694	2695	2696	2697	2698	2699	2700	2701	2702	2703	2704	2705	2706	2707	2708	2709	2710	2711	2712	2713	2714	2715	2716	2717	2718	2719	2720	2721	2722	2723	2724	2725	2726	2727
18	0AA8	2728	2729	2730	2731	2732	2733	2734	2735	2736	2737	2738	2739	2740	2741	2742	2743	2744	2745	2746	2747	2748	2749	2750	2751	2752	2753	2754	2755	2756	2757	2758	2759	2760	2761	2762	2763	2764	2765	2766	2767
19	0AD0	2768	2769	2770	2771	2772	2773	2774	2775	2776	2777	2778	2779	2780	2781	2782	2783	2784	2785	2786	2787	2788	2789	2790	2791	2792	2793	2794	2795	2796	2797	2798	2799	2800	2801	2802	2803	2804	2805	2806	2807
20	0AF8	2808	2809	2810	2811	2812	2813	2814	2815	2816	2817	2818	2819	2820	2821	2822	2823	2824	2825	2826	2827	2828	2829	2830	2831	2832	2833	2834	2835	2836	2837	2838	2839	2840	2841	2842	2843	2844	2845	2846	2847
21	0B20	2848	2849	2850	2851	2852	2853	2854	2855	2856	2857	2858	2859	2860	2861	2862	2863	2864	2865	2866	2867	2868	2869	2870	2871	2872	2873	2874	2875	2876	2877	2878	2879	2880	2881	2882	2883	2884	2885	2886	2887
22	0B48	2888	2889	2890	2891	2892	2893	2894	2895	2896	2897	2898	2899	2900	2901	2902	2903	2904	2905	2906	2907	2908	2909	2910	2911	2912	2913	2914	2915	2916	2917	2918	2919	2920	2921	2922	2923	2924	2925	2926	2927
23	0B70	2928	2929	2930	2931	2932	2933	2934	2935	2936	2937	2938	2939	2940	2941	2942	2943	2944	2945	2946	2947	2948	2949	2950	2951	2952	2953	2954	2955	2956	2957	2958	2959	2960	2961	2962	2963	2964	2965	2966	2967
24	0B98	2968	2969	2970	2971	2972	2973	2974	2975	2976	2977	2978	2979	2980	2981	2982	2983	2984	2985	2986	2987	2988	2989	2990	2991	2992	2993	2994	2995	2996	2997	2998	2999	3000	3001	3002	3003	3004	3005	3006	3007
25	0BC0	3008	3009	3010	3011	3012	3013	3014	3015	3016	3017	3018	3019	3020	3021	3022	3023	3024	3025	3026	3027	3028	3029	3030	3031	3032	3033	3034	3035	3036	3037	3038	3039	3040	3041	3042	3043	3044	3045	3046	3047

3047

True ASCII Conversion Table

Dec	x256	x256 +32768	Hex	CBM	True	Even Parity Dec	Even Parity Hex	Even Parity Oct	Binary	Odd Parity Dec	Odd Parity Hex	Odd Parity Oct	BCD	EBCDIC
0	**0**	**32768**	00		**NUL**	0	00	*000*	0/1 0000000	128	80	*200*	000000000	00
1	256	33024	01		**SOH**	129	81	*201*	1/0 0000001	1	01	*001*	000000001	01
2	512	33280	02		**STX**	130	82	*202*	1/0 0000010	2	02	*002*	000000010	02
3	768	33536	03		**ETX**	3	03	*003*	0/1 0000011	131	83	*203*	000000011	03
4	**1024**	**33792**	04		**EOT**	132	84	*204*	1/0 0000100	4	04	*004*	000000100	37
5	1280	34048	05		**ENQ**	5	05	*005*	0/1 0000101	133	85	*205*	000000101	2D
6	1536	34304	06		**ACK**	6	06	*006*	0/1 0000110	134	86	*206*	000000110	2E
7	1792	34560	07		**BEL**	135	87	*207*	1/0 0000111	7	07	*007*	000000111	2F
8	**2048**	**34816**	08		**BS**	136	88	*210*	1/0 0001000	8	08	*010*	000001000	16
9	2304	35072	09		**HT**	9	09	*011*	0/1 0001001	137	89	*211*	000001001	05
10	2560	35328	0A		**LF**	10	0A	*012*	0/1 0001010	138	8A	*212*	000010000	25
11	2816	35584	0B		**VT**	139	8B	*213*	1/0 0001011	11	0B	*013*	000010001	0B
12	**3072**	**35840**	0C		**FF**	12	0C	*014*	0/1 0001100	140	8C	*214*	000010010	0C
13	3328	36096	0D		**CR**	141	8D	*215*	1/0 0001101	13	0D	*015*	000010011	0D
14	3584	36352	0E		**SO**	142	8E	*216*	1/0 0001110	14	0E	*016*	000010100	0E
15	3840	36608	0F		**SI**	15	0F	*017*	0/1 0001111	143	8F	*217*	000010101	0F
16	**4096**	**36864**	10		**DLE**	144	90	*220*	1/0 0010000	16	10	*020*	000010110	10
17	4352	37120	11		**DC1**	17	11	*021*	0/1 0010001	145	91	*221*	000010111	11
18	4608	37376	12		**DC2**	18	12	*022*	0/1 0010010	146	92	*222*	000011000	12
19	4864	37632	13		**DC3**	147	93	*223*	1/0 0010011	19	13	*023.*	000011001	13
20	**5120**	**37888**	14		**DC4**	20	14	*024*	0/1 0010100	148	94	*224*	000100000	14
21	5376	38144	15		**NAK**	149	95	*225*	1/0 0010101	21	15	*025*	000100001	3D
22	5632	38400	16		**SYN**	150	96	*226*	1/0 0010110	22	16	*026*	000100010	32
23	5888	38656	17		**ETB**	23	17	*027*	0/1 0010111	151	97	*227*	000100011	26
24	**6144**	**38912**	18		**CAN**	24	18	*030*	0/1 0011000	152	98	*230*	000100100	18
25	6400	39168	19		**EM**	153	99	*231*	1/0 0011001	25	19	*031*	000100101	19
26	6656	39424	1A		**SUB**	154	9A	*232*	1/0 0011010	26	1A	*032*	000100110	3F
27	6912	39680	1B		**ESC**	27	1B	*033*	0/1 0011011	155	9B	*233*	000100111	27
28	**7168**	**39936**	1C		**FS**	156	9C	*234*	1/0 0011100	28	1C	*034*	000101000	22
29	7424	40192	1D		**GS**	29	1D	*035*	0/1 0011101	157	9D	*235*	000101001	
30	7680	40448	1E		**RS**	30	1E	*036*	0/1 0011110	158	9E	*236*	000110000	35
31	7936	40704	1F		**US**	159	9F	*237*	1/0 0011111	31	1F	*037*	000110001	
32	**8192**	**40960**	20			160	A0	*240*	1/0 0100000	32	20	*040*	000110010	40
33	8448	41216	21	**!**	**!**	33	21	*041*	0/1 0100001	161	A1	*241*	000110011	5A
34	8704	41472	22	**"**	**"**	34	22	*042*	0/1 0100010	162	A2	*242*	000110100	7F
35	8960	41728	23	**#**	**#**	163	A3	*243*	1/0 0100011	35	23	*043*	000110101	7B
36	**9216**	**41984**	24	**$**	**$**	36	24	*044*	0/1 0100100	164	A4	*244*	000110110	5B
37	9472	42240	25	**%**	**%**	165	A5	*245*	1/0 0100101	37	25	*045*	000110111	6C
38	9728	42496	26	**&**	**&**	166	A6	*246*	1/0 0100110	38	26	*046*	000111000	50
39	9984	42752	27	**'**	**'**	39	27	*047*	0/1 0100111	167	A7	*247*	000111001	7D
40	**10240**	**43008**	28	**(**	**(**	40	28	*050*	0/1 0101000	168	A8	*250*	001000000	4D
41	10496	43264	29	**)**	**)**	169	A9	*251*	1/0 0101001	41	29	*051*	001000001	5D
42	10752	43520	2A	*	*	170	AA	*252*	1/0 0101010	42	2A	*052*	001000010	5C
43	11008	43776	2B	**+**	**+**	43	2B	*053*	0/1 0101011	171	AB	*253*	001000011	4E
44	**11264**	**44032**	2C	**,**	**,**	172	AC	*254*	1/0 0101100	44	2C	*054*	001000100	6B
45	11520	44288	2D	**-**	**-**	45	2D	*055*	0/1 0101101	173	AD	*255*	001000101	60
46	11776	44544	2E	**.**	**.**	46	2E	*056*	0/1 0101110	174	AE	*256*	001000110	4B
47	12032	44800	2F	**/**	**/**	175	AF	*257*	1/0 0101111	47	2F	*057*	001000111	61
48	**12288**	**45056**	30	**0**	**0**	48	30	*060*	0/1 0110000	176	B0	*260*	001001000	F0
49	12544	45312	31	**1**	**1**	177	B1	*261*	1/0 0110001	49	31	*061*	001001001	F1
50	12800	45568	32	**2**	**2**	178	B2	*262*	1/0 0110010	50	32	*062*	001010000	F2
51	13056	45824	33	**3**	**3**	51	33	*063*	0/1 0110011	179	B3	*263*	001010001	F3
52	**13312**	**46080**	34	**4**	**4**	180	B4	*264*	1/0 0110100	52	34	*064*	001010010	F4
53	13568	46336	35	**5**	**5**	53	35	*065*	0/1 0110101	181	B5	*265*	001010011	F5
54	13824	46592	36	**6**	**6**	54	36	*066*	0/1 0110110	182	B6	*266*	001010100	F6
55	14080	46848	37	**7**	**7**	183	B7	*267*	1/0 0110111	55	37	*067*	001010101	F7
56	**14336**	**47104**	38	**8**	**8**	184	B8	*270*	1/0 0111000	56	38	*070*	001010110	F8
57	14592	47360	39	**9**	**9**	57	39	*071*	0/1 0111001	185	B9	*271*	001010111	F9
58	14848	47616	3A	**:**	**:**	58	3A	*072*	0/1 0111010	186	BA	*272*	001011000	7A
59	15104	47872	3B	**;**	**;**	187	BB	*273*	1/0 0111011	59	3B	*073*	001011001	5E
60	**15360**	**48128**	3C	<	<	60	3C	*074*	0/1 0111100	188	BC	*274*	001100000	4C
61	15616	48384	3D	=	=	189	BD	*275*	1/0 0111101	61	3D	*075*	001100001	7E
62	15872	48640	3E	>	>	190	BE	*276*	1/0 0111110	62	3E	*076*	001100010	6E
63	16128	48896	3F	**?**	**?**	63	3F	*077*	0/1 0111111	191	BF	*277*	001100011	6F

Even Parity: bit 7 OR'd in to make total number of bits Even
Odd Parity: bit 7 OR'd in to make total number of bits Odd

Dec	x256	x256 +32768	Hex	CBM	True	Even Parity Dec	Even Parity Hex	Even Parity Oct	Binary	Odd Parity Dec	Odd Parity Hex	Odd Parity Oct	BCD	EBCDIC
64	**16384**	**49152**	40	@	@	192	C0	300	1/0 1000000	64	40	100	001100100	7C
65	16640	49408	41	a	A	65	41	101	0/1 1000001	193	C1	301	001100101	C1
66	16896	49664	42	b	B	66	42	102	0/1 1000010	194	C2	302	001100110	C2
67	17152	49920	43	c	C	195	C3	303	1/0 1000011	67	43	103	001100111	C3
68	**17408**	**50176**	44	d	D	68	44	104	0/1 1000100	196	C4	304	001101000	C4
69	17664	50432	45	e	E	197	C5	305	1/0 1000101	69	45	105	001101001	C5
70	17920	50688	46	f	F	198	C6	306	1/0 1000110	70	46	106	001110000	C6
71	18176	50944	47	g	G	71	47	107	0/1 1000111	199	C7	307	001110001	C7
72	**18432**	**51200**	48	h	H	72	48	110	0/1 1001000	200	C8	310	001110010	C8
73	18688	51456	49	i	I	201	C9	311	1/0 1001001	73	49	111	001110011	C9
74	18944	51712	4A	j	J	202	CA	312	1/0 1001010	74	4A	112	001110100	D1
75	19200	51968	4B	k	K	75	4B	113	0/1 1001011	203	CB	313	001110101	D2
76	**19456**	**52224**	4C	l	L	204	CC	314	1/0 1001100	76	4C	114	001110110	D3
77	19712	52480	4D	m	M	77	4D	115	0/1 1001101	205	CD	315	001110111	D4
78	19968	52736	4E	n	N	78	4E	116	0/1 1001110	206	CE	316	001111000	D5
79	20224	52992	4F	o	O	207	CF	317	1/0 1001111	79	4F	117	001111001	D6
80	**20480**	**53248**	50	p	P	80	50	120	0/1 1010000	208	D0	320	010000000	D7
81	20736	53504	51	q	Q	209	D1	321	1/0 1010001	81	51	121	010000001	D8
82	20992	53760	52	r	R	210	D2	322	1/0 1010010	82	52	122	010000010	D9
83	21248	54016	53	s	S	83	53	123	0/1 1010011	211	D3	323	010000011	E2
84	**21504**	**54272**	54	t	T	212	D4	324	1/0 1010100	84	54	124	010000100	E3
85	21760	54528	55	u	U	85	55	125	0/1 1010101	213	D5	325	010000101	E4
86	22016	54784	56	v	V	86	56	126	0/1 1010110	214	D6	326	010000110	E5
87	22272	55040	57	w	W	215	D7	327	1/0 1010111	87	57	127	010000111	E6
88	**22528**	**55296**	58	x	X	216	D8	330	1/0 1011000	88	58	130	010001000	E7
89	22784	55552	59	y	Y	89	59	131	0/1 1011001	217	D9	331	010001001	E8
90	23040	55808	5A	z	Z	90	5A	132	0/1 1011010	218	DA	332	010010000	E9
91	23296	56064	5B	[	[	219	DB	333	1/0 1011011	91	5B	133	010010001	NA
92	**23552**	**56320**	5C	\	\	92	5C	134	0/1 1011100	220	DC	334	010010010	E0
93	23808	56576	5D	]	]	221	DD	335	1/0 1011101	93	5D	135	010010011	NA
94	24064	56832	5E	↑	↑	222	DE	336	1/0 1011110	94	5E	136	010010100	NA
95	24320	57088	5F	←	←	95	5F	137	0/1 1011111	223	DF	337	010010101	6D
96	**24576**	**57344**	60		`	96	60	140	0/1 1100000	224	E0	340	010010110	79
97	24832	57600	61		a	225	E1	341	1/0 1100001	97	61	141	010010111	81
98	25088	57856	62		b	226	E2	342	1/0 1100010	98	62	142	010011000	82
99	25344	58112	63		c	99	63	143	0/1 1100011	227	E3	343	010011001	83
100	**25600**	**58368**	64		d	228	E4	344	1/0 1100100	100	64	144	100000000	84
101	25856	58624	65		e	101	65	145	0/1 1100101	229	E5	345	100000001	85
102	26112	58880	66		f	102	66	146	0/1 1100110	230	E6	346	100000010	86
103	26368	59136	67		g	231	E7	347	1/0 1100111	103	67	147	100000011	87
104	**26624**	**59392**	68		h	232	E8	350	1/0 1101000	104	68	150	100000100	88
105	26880	59648	69		i	105	69	151	0/1 1101001	233	E9	351	100000101	89
106	27136	59904	6A		j	106	6A	152	0/1 1101010	234	EA	352	100000110	91
107	27392	60160	6B		k	235	EB	353	1/0 1101011	107	6B	153	100000111	92
108	**27648**	**60416**	6C		l	108	6C	154	0/1 1101100	236	EC	354	100001000	93
109	27904	60672	6D		m	237	ED	355	1/0 1101101	109	6D	155	100001001	94
110	28160	60928	6E		n	238	EE	356	1/0 1101110	110	6E	156	100010000	95
111	28416	61184	6F		o	111	6F	157	0/1 1101111	239	EF	357	100010001	96
112	**28672**	**61440**	70		p	240	F0	360	1/0 1110000	112	70	160	100010010	97
113	28928	61696	71		q	113	71	161	0/1 1110001	241	F1	361	100010011	98
114	29184	61952	72		r	114	72	162	0/1 1110010	242	F2	362	100010100	99
115	29440	62208	73		s	243	F3	363	1/0 1110011	115	73	163	100010101	A2
116	**29696**	**62464**	74		t	116	74	164	0/1 1110100	244	F4	364	100010110	A3
117	29952	62720	75		u	245	F5	365	1/0 1110101	117	75	165	100010111	A4
118	30208	62976	76		v	246	F6	366	1/0 1110110	118	76	166	100011000	A5
119	30464	63232	77		w	119	77	167	0/1 1110111	247	F7	367	100011001	A6
120	**30720**	**63488**	78		x	120	78	170	0/1 1111000	248	F8	370	100100000	A7
121	30976	63744	79		y	249	F9	371	1/0 1111001	121	79	171	100100001	A8
122	31232	64000	7A		z	250	FA	372	1/0 1111010	122	7A	172	100100010	A9
123	31488	64256	7B		{	123	7B	173	0/1 1111011	251	FB	373	100100011	C0
124	**31744**	**64512**	7C		\	252	FC	374	1/0 1111100	124	7C	174	100100100	6A
125	32000	64768	7D			125	7D	175	0/1 1111101	253	FD	375	100100101	D0
126	32256	65024	7E			126	7E	176	0/1 1111110	254	FE	376	100100110	A1
127	32512	65280	7F		DEL	255	FF	377	1/0 1111111	127	7F	177	100100111	07

Network Phone Numbers

Compuserve is offering a 30 minute free demonstration. To access the system, dial your local network that supports Compuserve. Once connected, type a carriage return.

When asked Host Name, type: CIS
When asked User ID, type: 77770,101
When asked Password, type: FREE-DEMO

The following letters are used to identify the network services.

C = CompuServe network
T = Tymnet network
G = GTE Telenet network
D = DataPac network

CANADA

Alberta (AB)

403-264-9340	D	Calgary
403-420-0185	D	Edmonton
403-791-2884	D	Fort McMurray
403-539-0100	D	Grande Prairie
403-329-8755	D	Lethbridge
403-526-6587	D	Medicine Hat
403-343-7200	D	Red Deer

British Columbia (BC)

604-374-5941	D	Kamloops
604-860-0331	D	Kelowna
604-354-4411	D	Nelson
604-564-4060	D	Prince George
604-635-7221	D	Terrace
604-687-6280	C	Vancouver
604-687-6138	C	Vancouver
604-687-6043	C	Vancouver
604-689-8601	D	Vancouver
604-388-9300	D	Victoria

Manitoba (MB)

204-725-0878	D	Brandon
204-638-9244	D	Dauphin
204-822-6237	D	Morden
204-239-1166	D	Portg la Prairi
204-785-8625	D	Selkirk
204-326-9826	D	Steinbach
204-778-4461	D	Thompson
204-475-2740	D	Winnipeg

New Brunswick (NB)

506-548-4461	D	Bathurst
506-759-8561	D	Campbellton
506-739-6621	D	Edmundston
506-454-9462	D	Fredericton
506-854-7078	D	Moncton
506-622-4451	D	Newcastle
506-693-7399	D	Saint John
506-328-9361	D	Woodstock

Newfoundland (NF)

709-726-4920	D	St. John's

Nova Scotia (NS)

902-667-5035	D	Amherst
902-543-6850	D	Bridgewater
902-477-2000	D	Halifax
902-678-1030	D	Kentville
902-752-0944	D	New Glasgow
902-539-7010	D	Sydney
902-662-3258	D	Truro

Ontario (ON)

416-791-8900	D	Brampton
519-756-0000	D	Brantford
613-345-0520	D	Brockville
613-589-2175	D	Chalk River
519-354-7710	D	Chatham
416-823-6000	D	Clarkson
613-938-9700	D	Cornwall
519-622-1714	D	Galt
416-523-6800	D	Hamilton
613-549-7720	D	Kingston
519-579-0009	D	Kitchner-Wtrloo
519-679-7500	D	London
416-357-2702	D	Niagara Falls
705-476-3900	D	North Bay
416-579-8920	D	Oshawa
613-567-9100	D	Ottawa
705-748-6940	D	Peterborough
519-336-9920	D	Sarnia
705-942-4960	D	Sault Ste.Marie
416-688-5620	D	St. Catharines
705-673-9602	D	Sudbury
807-623-9644	D	Thunder Bay
416-366-1869	C	Toronto
416-868-4000	D	Toronto
519-973-1000	D	Windsor
519-485-5220	D	Woodstock

Prince Edward Island (PE)

902-569-3391	D	Charlottetown

Province of Quebec (PQ)

819-477-7151	D	Drummondville
514-375-1240	D	Granby
514-759-8340	D	Joliette
418-545-2272	D	Jonquiere
514-878-0450	D	Montreal
418-647-4690	D	Quebec City
819-566-2770	D	Sherbrooke
514-743-3381	D	Sorel
514-744-9270	D	St. Hyacinthe
514-346-8779	D	St. Jean
514-432-3453	D	St. Jerome
819-373-2600	D	Trois Rivieres
514-377-1260	D	Valleyfield

Saskatchewan (SA)

306-693-7611	D	Moose Jaw
306-922-4233	D	Prince Albert
306-565-0111	D	Regina
306-665-6660	D	Saskatoon

USA

Alaska (AK)

907-276-0271	G	Anchorage
907-338-7222	T	Anchorage
907-456-3282	T	Fairbanks
907-586-9700	G	Juneau
907-789-7009	T	Juneau
907-659-2777	T	Prudhoe Bay

Alabama (AL)

205-236-2655	T	Anniston
205-328-2310	G	Bessemer
205-879-2250	C	Birmingham
205-879-2280	C	Birmingham
205-328-2310	G	Birmingham
205-942-4141	T	Birmingham
205-792-0914	T	Dothan
205-767-7960	G	Florence
205-536-4405	C	Huntsville
205-539-2281	G	Huntsville
205-882-3003	T	Huntsville
205-432-1680	G	Mobile
205-343-8414	T	Mobile
205-262-0010	C	Montgomery
205-269-0090	G	Montgomery
205-265-4570	T	Montgomery
205-767-7960 .	?	Sheffield

Arkansas (AR)

501-782-3210	T	Ft. Smith
501-321-9741	T	Hot Springs
501-932-1147	T	Jonesboro
501-666-8464	C	Little Rock
501-666-8478	C	Little Rock
501-372-4616	G	Little Rock
501-666-6886	T	Little Rock
501-756-2201	T	Springdale

Arizona (AZ)

602-256-2951	C	Mesa
602-254-0244	G	Mesa
602-256-2951	C	Phoenix
602-254-0244	G	Phoenix
602-254-5811	T	Phoenix
602-256-2951	C	Scottsdale
602-254-0244	G	Scottsdale
602-256-2951	C	Tempe
602-254-0244	G	Tempe
602-748-2004	C	Tucson
602-748-2009	C	Tucson
602-747-0107	G	Tucson
602-790-0764	T	Tucson

California (CA)

213-507-0909	G	Alhambra
818-308-1800	T	Alhambra
714-520-9724	C	Anaheim
714-520-9733	C	Anaheim
714-558-6061	G	Anaheim
714-966-0313	T	Anaheim
415-778-3420	T	Antioch
818-308-1800	T	Arcadia
805-323-7691	C	Bakersfield
805-327-8146	G	Bakersfield
805-325-8366	T	Bakersfield
415-366-1092	T	Belmont
818-789-9002	T	Beverly Hills
818-841-7890	T	Burbank
415-591-0726	G	Burlingame
415-952-4757	T	Burlingame
408-980-8100	T	Campbell
213-306-2984	G	Canoga Park
818-789-9002	T	Canoga Park
415-581-2631	C	Castro Valley
916-893-1876	T	Chico
714-824-9000	G	Colton
714-370-1200	T	Colton
213-516-1007	G	Compton
415-676-2834	G	Concord
415-682-3851	T	Concord
714-371-2291	T	Corona
213-330-1630	G	Covina
714-594-4567	T	Covina
213-390-9617	C	Culver City
408-249-5361	C	Cupertino
408-294-9119	G	Cupertino
408-980-8100	T	Cupertino
916-753-3722	T	Davis
714-594-4567	T	Diamond Bar
213-507-0909	G	El Monte
213-640-1281	T	El Segundo
619-741-7756	G	Escondido
619-941-6700	T	Escondido
707-445-3281	T	Eureka
415-490-7366	T	Fremont
209-252-1892	C	Fresno
209-233-0961	G	Fresno
209-442-4328	T	Fresno
714-558-6061	G	Fullerton
714-898-9820	G	Garden Grove
714-966-0313	T	Garden Grove
818-507-0909	G	Glendale
415-881-1382	G	Hayward
415-430-2900	T	Hayward
213-937-3580	G	Hollywood
213-689-9040	G	Hollywood
714-558-6061	G	Huntington Bch
213-937-3580	G	Inglewood
213-689-9040	G	Inglewood
714-851-9612	C	Irvine
714-756-8341	T	Irvine
805-945-7841	T	Lancaster
213-591-8392	C	Long Beach
213-548-6141	G	Long Beach
213-435-0900	T	Long Beach
408-249-5361	C	Los Altos
415-856-9995	G	Los Altos
408-980-8100	T	Los Altos
213-739-8906	C	Los Angeles
213-739-0371	C	Los Angeles
213-937-3580	G	Los Angeles
213-689-9040	G	Los Angeles
213-626-2400	T	Los Angeles
805-985-7843	T	Mantea
213-821-2257	T	Mar Vista
213-306-2984	G	Marina Del Rey
213-821-2257	T	Marina Del Rey
415-366-1092	T	Menlo Park
818-789-9002	T	Mission Hills
209-576-2852	G	Modesto
209-571-0408	T	Modesto
408-375-2675	G	Monterey
408-988-8762	C	Mt. View
415-856-9995	G	Mt. View
408-980-8100	T	Mt. View
818-982-1813	C	N. Hollywood
707-257-2656	T	Napa
714-851-9612	C	Newport Beach
714-558-6061	G	Newport Beach
714-756-8341	T	Newport Beach
818-789-9002	T	Northridge
213-404-2237	G	Norwalk
213-435-0900	T	Norwalk
415-836-4911	G	Oakland
415-430-2900	T	Oakland
714-594-4567	T	Ontario
805-656-6760	G	Oxnard
805-985-7843	T	Oxnard
619-320-0772	T	Palm Springs
415-591-5591	C	Palo Alto
415-591-5846	C	Palo Alto
415-856-9995	G	Palo Alto
415-366-1092	T	Palo Alto
213-507-0909	G	Pasadena
818-308-1800	T	Pasadena
415-682-3851	T	Pleasant Hill
415-846-0828	C	Pleasanton
415-462-8900	T	Pleasanton
714-623-2651	C	Pomona
714-594-4567	T	Pomona
805-985-7843	T	Port Huenene
619-487-6648	C	Rancho Bernardo
619-485-1990	T	Rancho Bernardo
916-223-0449	T	Redding
415-591-0726	G	Redwood City
415-366-1092	T	Redwood City
714-359-7801	C	Riverside
714-824-9000	G	Riverside
714-370-1200	T	Riverside
916-971-4681	C	Sacramento
916-448-6262	G	Sacramento
916-448-4300	T	Sacramento
408-443-4940	G	Salinas
408-443-4333	T	Salinas
714-381-3469	C	San Bernadino
714-824-9000	G	San Bernadino
714-370-1200	T	San Bernadino
415-952-4757	T	San Bruno
415-591-0726	G	San Carlos
415-366-1092	T	San Carlos
714-498-9504	T	San Clemente
619-283-6021	C	San Diego
619-283-6091	C	San Diego
619-231-1922	G	San Diego
619-296-3370	T	San Diego
818-789-9002	T	San Fernando
415-956-4281	C	San Francisco
415-956-4191	C	San Francisco
415-362-6200	G	San Francisco
415-974-1300	T	San Francisco
408-249-5361	C	San Jose
408-249-5472	C	San Jose
408-294-9119	G	San Jose
408-980-8100	T	San Jose
805-546-8541	T	San Luis Obispo
415-591-5846	C	San Mateo
415-591-5591	C	San Mateo
415-591-0726	G	San Mateo
213-548-6141	G	San Pedro
213-435-0900	T	San Pedro
415-492-0752	G	San Rafael
415-492-9320	T	San Rafael
714-558-6061	G	Santa Ana
714-966-0313	T	Santa Ana
805-682-5361	G	Santa Barbara
805-963-9241	T	Santa Barbara
408-988-8762	C	Santa Clara
408-294-9119	G	Santa Clara
408-980-8100	T	Santa Clara
408-425-8455	G	Santa Cruz
408-475-0981	T	Santa Cruz
213-306-2984	G	Santa Monica
213-821-2257	T	Santa Monica
707-578-9325	G	Santa Rosa
707-527-6180	T	Santa Rosa
818-789-9002	T	Sherman Oaks
818-355-4816	C	Sierra Madre
209-465-7251	C	Stockton
209-473-2056	G	Stockton
209-467-0601	T	Stockton
408-294-9119	G	Sunnyvale
408-980-8100	T	Sunnyvale
805-499-0388	C	Thousand Oaks
805-499-0371	C	Thousand Oaks
805-496-3473	T	Thousand Oaks
213-542-4311	C	Torrance
213-548-6141	G	Torrance
707-557-0333	T	Vallejo
818-902-0932	C	Van Nuys
818-902-0934	C	Van Nuys
818-789-9002	T	Van Nuys
805-656-6760	G	Ventura
805-985-7843	T	Ventura
209-625-5523	T	Visalia
619-941-6700	T	Vista
415-938-9550	T	Walnut Creek
714-594-4567	T	West Covina
818-887-3160	G	Woodland Hills
415-856-9995	G	Woodside

Colorado (CO)

303-629-5563	C	Aurora
303-337-6000	G	Aurora
303-629-5563	C	Boulder
303-337-6000	G	Boulder
303-830-9210	T	Boulder
303-596-0910	C	Colorado Sprngs
303-635-5361	G	Colorado Sprngs
303-590-1003	T	Colorado Sprngs
303-629-5563	C	Denver
303-629-0668	C	Denver
303-337-6000	G	Denver
303-830-9210	T	Denver
303-221-0687	T	Fort Collins
303-241-1885	C	Grand Junction
303-241-1889	C	Grand Junction
303-356-0425	T	Greeley
303-629-5563	C	Lakewood
303-337-6000	G	Lakewood
303-543-3313	T	Pueblo

Connecticut (CT)

203-242-7140	T	Bloomfield
203-926-0001	C	Bridgeport
203-335-5055	G	Bridgeport
203-367-6021	T	Bridgeport
203-797-0467	C	Danbury
203-794-9075	G	Danbury
203-797-9539	T	Danbury
203-965-0000	T	Darien
203-226-5250	T	Fairfield
203-348-0787	G	Greenwich
203-236-5931	C	Hartford
203-236-2581	C	Hartford
203-247-9479	G	Hartford
203-242-7140	T	Hartford
203-235-5180	T	Meriden
203-624-5954	G	Milford
203-624-5954	G	New Haven
203-773-0082	T	New Haven
203-444-1709	T	New London
203-773-0082	T	North Haven
203-226-5250	T	Norwalk
203-444-1709	T	Norwich
203-967-4589	C	Stamford
203-348-0787	G	Stamford
203-965-0000	T	Stamford
203-574-0500	C	Waterbury
203-753-4512	G	Waterbury
203-755-5994	T	Waterbury
203-247-9479	G	West Hartford
203-773-0082	T	West Haven
203-222-1748	C	Westport
203-226-2704	C	Westport
203-226-5250	T	Westport

District of Columbia (DC)

703-352-7500	C	Washington
703-841-9834	C	Washington
202-429-7896	G	Washington
703-691-8390	T	Washington
703-691-8200	T	Washington

Delaware (DE)

302-678-0449	T	Dover
302-652-8732	C	Wilmington
302-454-7710	G	Wilmington
302-652-2060	T	Wilmington

Florida (FL)

305-368-8300	G	Boca Raton
305-395-7330	T	Boca Raton
813-323-4026	G	Clearwater
813-796-2166	T	Clearwater
904-252-9914	G	Daytona Beach
904-255-4783	T	Daytona Beach
305-771-8074	C	Ft. Lauderdale
305-772-3240	C	Ft. Lauderdale
305-764-4505	G	Ft. Lauderdale
305-463-0882	T	Ft. Lauderdale
813-337-0308	G	Ft. Myers
813-936-4221	T	Ft. Myers
305-466-0661	T	Ft. Pierce
904-377-3005	G	Gainesville
904-376-0939	T	Gainesville
305-463-0882	T	Hollywood
904-246-9961	C	Jacksonville
904-241-8191	C	Jacksonville
904-353-1818	G	Jacksonville
904-721-8100	T	Jacksonville
813-688-4366	G	Lakeland
813-688-5776	T	Lakeland
305-841-0020	T	Longwood
305-676-4336	T	Melbourne
305-459-0671	T	Merritt Isle
305-667-3564	C	Miami
305-665-6425	C	Miami
305-372-0230	G	Miami
305-624-7900	T	Miami
904-351-0070	T	Ocala
305-273-8780	C	Orlando
305-273-8805	C	Orlando
305-422-4088	G	Orlando
305-841-0020	T	Orlando
305-723-2353	C	Palm Bay
904-769-9446	T	Panama City
904-438-4562	G	Pensacola
904-477-3344	T	Pensacola
305-941-5445	G	Pompano Beach
813-921-3369	G	Sarasota
813-365-6980	T	Sarasota
813-323-4026	G	St. Petersburg
813-796-2166	T	St. Petersburg
904-224-6021	C	Tallahassee
904-222-4144	C	Tallahassee
904-681-1902	G	Tallahassee
904-878-2267	T	Tallahassee
813-875-0633	C	Tampa
813-224-9920	G	Tampa
813-932-7070	T	Tampa
305-833-6691	G	West Palm Beach
305-471-9310	T	West Palm Beach
813-688-5776	T	Winterhaven

Georgia (GA)

912-883-2246	T	Albany
404-549-4524	G	Athens
404-546-0167	T	Athens
404-237-8113	C	Atlanta
404-237-3003	C	Atlanta
404-577-8911	G	Atlanta
404-446-0270	T	Atlanta

404-733-0346	C	Augusta
404-790-4119	G	Augusta
404-722-7967	T	Augusta
404-571-0556	G	Columbus
404-327-0396	T	Columbus
912-741-1011	G	Macon
912-744-0605	T	Macon
404-424-0025	T	Marietta
404-291-1000	T	Rome
912-236-2605	G	Savannah
912-232-6751	T	Savannah

Hawaii (HI)

808-524-8110	G	Honolulu
808-528-4450	T	Honolulu

Iowa (IA)

319-364-0911	G	Cedar Rapids
319-363-7514	T	Cedar Rapids
402-341-7733	G	Council Bluffs
319-324-2445	G	Davenport
309-794-0731	T	Davenport
515-270-9410	C	Des Moines
515-270-1581	C	Des Moines
515-288-4403	G	Des Moines
515-277-7752	T	Des Moines
319-556-8263	T	Dubuque
319-351-1421	G	Iowa City
319-354-7371	T	Iowa City
515-753-0667	T	Marshalltown
712-252-1681	T	Sioux City
319-233-9227	T	Waterloo

Idaho (ID)

208-384-5660	C	Boise
208-384-5666	C	Boise
208-343-0611	G	Boise
208-343-0404	T	Boise
208-523-2964	T	Idaho Falls
208-233-2501	T	Pocatello

Illinois (IL)

312-938-0500	G	Arlington Hghts
312-896-2137	C	Aurora
312-859-8483	G	Aurora
312-859-1143	T	Aurora
618-277-9806	T	Belleville
217-384-6428	G	Champaign
217-356-7552	T	Champaign
312-443-1250	C	Chicago
312-332-7382	C	Chicago
312-938-0500	G	Chicago
312-922-4601	T	Chicago
312-938-0500	G	Cicero
217-431-3133	T	Danville
217-422-0835	G	Decatur
217-422-0612	T	Decatur
312-790-4400	T	Downers Grove
314-421-4990	G	East St. Louis
312-771-9667	T	Forest Park
815-233-5585	T	Freeport
312-790-4400	T	Glen Ellyn
815-722-0703	G	Joliet
815-727-1019	T	Joliet
815-932-0850	T	Kankakee
312-438-3771	T	Lake Zurick
312-362-0820	T	Libertyville
312-953-9680	C	Lombard
219-838-6353	T	Merrillville
312-938-0500	G	Oak Park
312-932-7370	C	Oakbrook Terr.
309-637-8570	G	Peoria
309-637-5961	T	Peoria
309-794-0731	T	Rock Island
815-965-0400	G	Rockford
815-398-6090	T	Rockford
312-938-0500	G	Skokie
217-522-5101	C	Springfield
217-753-1373	G	Springfield
217-753-7905	T	Springfield
312-859-1143	T	St. Charles
217-384-6428	G	Urbana
217-356-7552	T	Urbana
312-790-4400	T	Wheaton

Indiana (IN)

812-332-1344	G	Bloomington
812-424-7693	G	Evansville
812-464-8181	T	Evansville
219-447-0573	C	Ft. Wayne
219-426-2268	G	Ft. Wayne
219-422-2581	T	Ft. Wayne
219-882-8800	G	Gary
219-838-6353	T	Highland
317-638-2517	C	Indianapolis
317-638-2762	C	Indianapolis
317-635-9630	G	Indianapolis
317-257-3461	T	Indianapolis
317-455-2460	G	Kokomo
317-452-8241	T	Kokomo
317-742-1165	G	Lafayette
317-742-0189	T	Lafayette
317-664-9033	T	Marion
219-233-7104	G	Mishawka
317-284-4474	T	Muncie
219-674-5171	C	Osceola
219-233-7104	G	Osceola
219-233-7104	G	South Bend
219-234-5005	T	South Bend
812-234-8429	G	Terre Haute
812-232-3605	T	Terre Haute

Kansas (KS)

816-221-9900	G	Kansas City
913-384-1544	T	Kansas City
913-749-0271	T	Lawrence
913-682-2660	T	Leavenworth
913-776-5189	T	Manhatten
913-384-1544	T	Mission
913-823-7186	T	Salina
913-384-1544	T	Shawnee Mission
913-233-9880	G	Topeka
913-233-1682	T	Topeka
316-689-8765	C	Wichita
316-262-5669	G	Wichita
316-265-1241	T	Wichita

Kentucky (KY)

502-782-7941	G	Bowling Green
502-782-0436	T	Bowling Green
502-875-4654	G	Frankfort
606-259-3446	C	Lexington
606-233-0312	G	Lexington
606-253-3463	T	Lexington
502-581-9526	C	Louisville
502-589-5580	G	Louisville
502-499-7110	T	Louisville
502-685-1318	T	Owensboro

Louisiana (LA)

318-443-9544	T	Alexandria
504-273-0184	C	Baton Rouge
504-343-0753	G	Baton Rouge
504-924-5102	T	Baton Rouge
318-234-1095	G	Lafayette
318-237-9500	T	Lafayette
318-436-1633	T	Lake Charles
318-387-0879	C	Monroe
318-387-6330	G	Monroe
318-322-4109	T	Monroe
504-948-9542	C	New Orleans
504-949-2086	C	New Orleans
504-524-4094	G	New Orleans
504-524-4371	T	New Orleans
318-424-5380	C	Shreveport
318-221-5833	G	Shreveport
318-688-5840	T	Shreveport

Massachusetts (MA)

413-256-8194	C	Amherst
617-292-0600	G	Arlington
617-226-4471	T	Attleboro
617-267-2569	C	Boston
617-292-0600	G	Boston
617-292-1900	T	Boston
617-586-9803	C	Brockton
617-584-6873	T	Brockton
617-292-0600	G	Brookline
617-272-3615	C	Burlington
617-267-2569	C	Cambridge
617-292-0600	G	Cambridge
617-292-1900	T	Cambridge
413-781-3811	G	Chicopee
617-371-0354	C	Concord
617-675-1750	T	Fall River
617-343-8480	T	Fitchburg
617-875-3814	C	Framingham
617-620-1264	T	Framingham
617-352-2328	C	Georgetown
413-781-3811	G	Holyoke
617-568-8019	C	Hudson
617-681-8802	T	Lawrence
617-863-1550	G	Lexington
617-452-0819	T	Lowell
617-897-4779	C	Maynard
617-359-7603	C	Medfield
617-292-0600	G	Medford
617-533-2722	C	Medway
617-478-0653	C	Mendon
617-996-8596	T	New Bedford
617-267-2569	C	Newton
617-292-0600	G	Newton
413-442-6965	T	Pittsfield
617-267-2569	C	Quincy
617-292-0600	G	Quincy
617-292-0600	G	Somerville
413-734-7362	C	Springfield
413-781-3811	G	Springfield
413-781-6830	T	Springfield
617-822-7799	T	Taunton
617-890-0232	C	Waltham
617-292-0600	G	Waltham
617-366-1577	C	Westboro
617-935-2057	T	Woburn
617-540-7500	G	Woods Hole
617-793-9839	C	Worcester
617-755-4740	G	Worcester
617-791-9000	T	Worcester

Maryland (MD)

301-272-3800	T	Aberdeen
301-224-8550	G	Annapolis
301-254-7113	C	Baltimore
301-962-5010	G	Baltimore
301-547-8100	T	Baltimore
202-429-7896	G	Bethesda
301-652-0800	T	Chevy Chase
301-722-7710	T	Cumberland
301-962-5010	G	Dundalk
301-293-1072	T	Frederick
301-293-1072	T	Hagerstown
301-559-0200	C	Hyattsville
301-293-1072	T	Myersville
202-429-7896	G	Rockville
301-652-0800	T	Rockville
202-429-7896	G	Silver Spring
301-962-5010	G	Towson

Maine (ME)

207-786-0645	T	Auburn
207-622-3123	G	Augusta
207-947-1196	T	Bangor
207-947-1196	T	Brewer
207-236-8505	C	Camden
207-786-0645	T	Lewiston
207-773-4219	G	Portland
207-775-5971	T	Portland

Michigan (MI)

313-761-1202	C	Ann Arbor
313-996-5995	G	Ann Arbor
313-662-8282	T	Ann Arbor
616-968-0929	G	Battle Creek
616-962-1851	T	Battle Creek
616-925-3134	T	Benton Hbr/St.J
616-775-6089	T	Cadillac
313-567-3405	C	Detroit
313-567-4910	C	Detroit
313-964-5538	G	Detroit
313-962-2870	T	Detroit
517-321-2388	C	East Lansing
313-238-6202	C	Flint
313-235-8517	G	Flint
313-732-7303	T	Flint
517-695-6751	T	Freeland
616-774-0966	G	Grand Rapids
616-459-2304	T	Grand Rapids
517-789-8133	T	Jackson
517-782-0584	T	Jackson
616-344-2298	C	Kalamazoo
616-344-5312	C	Kalamazoo
616-345-3088	G	Kalamazoo
616-388-2130	T	Kalamazoo
517-321-2388	C	Lansing
517-484-0062	G	Lansing
517-482-5721	T	Lansing
616-723-6071	T	Manistee
517-695-6751	T	Midland
616-725-8136	T	Muskegon
313-459-8900	T	Plymouth
313-985-6005	T	Port Huron
517-893-1161	C	Saginaw
517-790-5166	G	Saginaw
517-695-6751	T	Saginaw
313-827-4710	G	Southfield
313-424-8024	T	Southfield
616-925-3134	T	St.Joe/Benton H
616-947-0050	T	Traverse City
313-362-2540	C	Troy
313-575-9152	G	Warren

Minnesota (MN)

218-722-1719	G	Duluth
218-722-7441	T	Duluth
507-625-9481	T	Mankato
612-342-2207	C	Minneapolis
612-341-2459	G	Minneapolis
612-333-2799	T	Minneapolis
507-289-1900	T	Rochester
612-252-9093	T	St. Cloud
612-341-2459	G	St. Paul
612-333-2799	T	St. Paul

Missouri (MO)

314-731-8002	T	Bridgeton
314-875-1290	T	Columbia
314-421-4990	G	Florissant
314-634-5178	G	Jefferson City
314-634-8323	T	Jefferson City
417-782-3037	T	Joplin
816-474-3770	C	Kansas City
816-221-9900	G	Kansas City
913-384-1544	T	Kansas City
314-364-3486	T	Rolla
417-864-4814	G	Springfield
417-831-5044	T	Springfield
816-232-1897	T	St. Joseph
314-241-3101	C	St. Louis
314-241-3102	C	St. Louis
314-421-4990	G	St. Louis
314-731-8002	T	St. Louis

Mississippi (MS)

601-982-0463	C	Jackson
601-969-0036	G	Jackson
601-355-9741	T	Jackson
601-693-8216	T	Meridian
601-769-6502	T	Pascagoula
601-769-6673	T	Pascagoula
601-634-6670	T	Vicksburg

Montana (MT)

406-245-7649	G	Billings
406-252-4880	T	Billings
406-586-7638	T	Bozeman
406-494-6615	T	Butte
406-727-0100	T	Great Falls
406-443-0000	G	Helena
406-721-5900	G	Missoula
406-728-2415	T	Missoula

North Carolina (NC)

704-252-9134	G	Asheville
704-253-3873	T	Asheville
704-333-6654	C	Charlotte
704-333-7155	C	Charlotte
704-332-3131	G	Charlotte
704-376-2545	T	Charlotte
704-376-2544	T	Charlotte
919-549-8139	G	Davidson
919-549-8139	G	Durham
919-549-8952	T	Durham
919-323-4501	G	Fayetteville
919-323-4202	T	Fayetteville
919-373-1635	C	Greensboro
919-273-2851	G	Greensboro
919-273-0332	T	Greensboro
919-758-7854	T	Greenville
919-889-2253	G	High Point
919-882-6858	T	High Point
919-878-8570	C	Raleigh
919-549-8139	G	Raleigh
919-829-0536	T	Raleigh
919-549-8139	G	Research TriPrk
919-343-0770	T	Wilmington
919-725-2126	G	Winston-Salem
919-761-1103	T	Winston-Salem

North Dakota (ND)

701-223-6839	T	Bismark
701-280-0210	T	Fargo
701-775-0531	T	Grand Forks
701-663-2256	G	Mandan
701-838-1114	T	Minot

Nebraska (NE)

402-475-4964	G	Lincoln
402-475-8659	T	Lincoln
402-895-5288	C	Omaha
402-341-7733	G	Omaha
402-397-0414	T	Omaha

New Hampshire (NH)

603-224-1024	G	Concord
603-668-1420	G	Manchester
603-623-0409	T	Manchester
603-883-0884	C	Merrimack
603-883-5551	C	Nashua
603-889-8618	G	Nashua
603-882-0435	T	Nashua
603-431-2302	G	Portsmouth
603-893-6200	T	Salem

New Jersey (NJ)

609-348-0561	G	Atlantic City
609-345-6888	T	Atlantic City
201-623-6818	G	Bayonne
609-665-7893	C	Cherry Hill
609-665-6244	C	Cherry Hill
609-665-5600	T	Cherry Hill
201-542-2180	T	Eatontown
201-894-8250	T	Englewood Cliff
201-968-9000	C	Greenbrook
201-852-8070	C	Hackettstown
201-852-8502	C	Hackettstown
201-623-6818	G	Jersey City
201-432-4907	T	Jersey City
201-460-0100	T	Lyndhurst
201-460-0180	T	Lyndhurst
609-596-1500	G	Marlton
201-783-5400	C	Montclair
609-665-5600	T	Moorestown
201-455-0275	G	Morristown
201-539-1222	T	Morristown
201-745-2900	G	New Brunswick
201-484-2275	C	Newark
201-623-6818	G	Newark
201-824-1212	T	Newark
201-898-1935	C	Parsippany
201-778-5600	G	Passaic
201-684-7560	G	Paterson
609-665-5600	T	Pennsaukin
201-981-1900	T	Piscataway
609-683-4770	C	Princeton
609-683-4776	C	Princeton
609-799-5587	G	Princeton
609-452-1018	T	Princeton
201-445-8346	T	Ridgewood
609-989-8847	G	Trenton
609-989-8480	T	Trenton
201-824-1212	T	Union
201-623-6818	G	Union City
201-785-4480	T	Wayne

New Mexico (NM)

505-345-3631	C	Albuquerque
505-243-4479	G	Albuquerque
505-242-8344	T	Albuquerque
505-524-1944	T	Las Cruces
505-988-5953	T	Santa Fe

Nevada (NV)

702-293-0300	T	Boulder City
702-885-8411	T	Carson City
702-878-0056	C	Las Vegas
702-737-6861	G	Las Vegas
702-293-0300	T	Las Vegas
702-786-5308	C	Reno
702-786-5356	C	Reno
702-827-6900	G	Reno
702-885-8411	T	Reno

New York (NY)

518-439-7491	C	Albany
518-465-8444	G	Albany
518-458-8300	T	Albany
607-772-1225	C	Binghamton
607-772-6642	G	Binghamton
607-772-1153	T	Binghamton
716-874-3751	C	Buffalo
716-847-1440	G	Buffalo
716-845-6610	T	Buffalo
607-962-4481	T	Corning
516-667-5566	G	Deer Park
607-737-9010	T	Elmira
516-292-0320	G	Hempstead
516-485-7422	T	Hempstead
516-681-7240	C	Hicksville
516-681-7347	C	Hicksville
516-420-1221	T	Huntington
607-257-6601	T	Ithaca
516-420-1221	T	Melville
516-294-3120	T	Mineola
212-422-8820	C	New York
212-758-4114	C	New York
212-758-2090	C	New York
212-736-0099	G	New York
212-785-5400	T	New York
212-269-6985	T	New York
212-685-4414	T	New York
212-532-0437	T	New York
716-285-2561	T	Niagara Falls
914-473-2240	G	Poughkeepsie
914-473-0401	T	Poughkeepsie
716-458-3460	C	Rochester
716-458-3465	C	Rochester
716-454-3430	G	Rochester
716-248-8000	T	Rochester
516-467-5178	T	Ronkonkoma
518-465-8444	G	Schenectady
516-732-2198	C	Selden
315-463-6512	C	Syracuse
315-472-5583	G	Syracuse
315-437-7111	T	Syracuse
716-694-6263	C	Tonawanda
518-465-8444	G	Troy
315-735-2291	T	Utica
315-797-0920	G	Utica/Rome
914-428-9270	C	White Plains
914-949-4510	C	White Plains
914-428-9335	C	White Plains
914-328-9199	G	White Plains
914-684-6075	T	White Plains
516-877-2862	C	Williston Park

Ohio (OH)

216-867-1243	C	Akron
216-867-1237	C	Akron
216-678-5115	G	Akron
216-535-1861	T	Akron
614-594-8364	C	Athens
216-455-2516	C	Canton
216-455-2126	C	Canton
216-452-0903	G	Canton
216-455-0066	T	Canton
513-721-2691	C	Cincinnati
513-579-0908	C	Cincinnati
513-579-0390	G	Cincinnati
513-489-2100	T	Cincinnati
216-771-0723	C	Cleveland
216-771-6860	C	Cleveland
216-575-1658	G	Cleveland
216-241-0024	T	Cleveland
614-457-2105	C	Columbus
614-463-9340	G	Columbus
614-221-1862	T	Columbus
513-461-1064	C	Dayton
513-461-5254	G	Dayton
513-223-3847	T	Dayton
216-323-5059	G	Elyria
216-575-1658	G	Euclid

614-587-0932	C	Granville
513-894-1521	T	Hamilton
216-678-5115	G	Kent
419-224-2998	T	Lima
419-526-6067	T	Mansfield
513-644-0096	T	Marysville
216-455-0066	T	North Canton
216-575-1658	G	Parma
513-324-3816	T	Springfield
419-255-8116	C	Toledo
419-255-7881	G	Toledo
419-255-7790	T	Toledo
216-394-6529	T	Warren
216-743-1296	G	Youngstown
216-744-5326	T	Youngstown

Oklahoma (OK)

405-223-1552	T	Ardmore
405-232-4546	G	Bethany
405-233-7903	T	Enid
405-355-0745	T	Lawton
405-232-4546	G	Norman
405-946-4799	C	Oklahoma City
405-946-4860	C	Oklahoma City
405-232-4546	G	Oklahoma City
405-947-6387	T	Oklahoma City
405-624-1112	G	Stillwater
918-749-8801	C	Tulsa
918-749-8850	C	Tulsa
918-584-3247	G	Tulsa
918-582-4433	T	Tulsa

Oregon (OR)

503-754-9273	G	Corvallis
503-683-1460	G	Eugene
503-485-0027	T	Eugene
503-779-6343	G	Medford
503-773-1257	T	Medford
503-232-1072	C	Portland
503-232-4026	C	Portland
503-295-3028	G	Portland
503-226-0627	T	Portland
503-378-7712	G	Salem
503-399-1453	T	Salem

Pennsylvania (PA)

215-776-6960	C	Allentown
215-435-3330	G	Allentown
215-865-6978	T	Allentown
814-946-8888	T	Altoona
215-865-6978	T	Bethlehem
215-873-0300	T	Downington
814-453-7538	C	Erie
814-899-2241	G	Erie
814-456-8501	T	Erie
412-837-3800	T	Greensburg
717-657-9633	C	Harrisburg
717-236-6882	G	Harrisburg
717-763-6481	T	Harrisburg
814-535-7576	G	Johnstown
215-265-7230	C	King of Prussia
215-337-4300	G	King of Prussia
215-337-9900	T	King of Prussia
717-397-7731	T	Lancaster
412-837-3800	T	Latrobe
215-736-0495	T	Levittown
412-652-4223	T	New Castle
215-666-9190	T	Norristown
412-288-9950	G	Penn Hills
215-563-1051	C	Philadelphia
215-574-0620	G	Philadelphia
215-567-4390	T	Philadelphia
412-391-8818	C	Pittsburgh
412-391-7732	C	Pittsburgh
412-288-9950	G	Pittsburgh
412-642-6778	T	Pittsburgh
215-374-5600	C	Reading
215-372-4473	T	Reading
717-961-5321	G	Scranton
717-346-4516	T	Scranton
814-237-6408	T	State College
215-574-0620	G	Upper Darby
215-666-9190	T	Valley Forge
717-822-1272	T	Wilkes Barre
717-846-6550	G	York
717-846-3900	T	York

Puerto Rico (PR)

800-462-4213	T	Mayaquez
800-462-4213	T	Ponce
809-792-5900	T	San Juan

Rhode Island (RI)

401-847-0502	T	Newport
401-273-0200	T	Pawtucket
401-781-8500	C	Providence
401-781-8505	C	Providence
401-751-7912	G	Providence
401-273-0200	T	Providence
401-751-7912	G	Warwick
401-765-2400	T	Woonsocket

South Carolina (SC)

803-763-0090	C	Charleston
803-722-4303	G	Charleston
803-577-0452	T	Charleston
803-798-3630	C	Columbia
803-254-0695	G	Columbia
803-254-7563	T	Columbia
803-233-3486	G	Greenville
803-271-9213	T	Greenville
803-585-1637	G	Spartanburg
803-582-7924	T	Spartanburg

South Dakota (SD)

605-224-0481	G	Pierre
605-341-3733	C	Rapid City
605-341-5337	T	Rapid City
605-336-8593	G	Sioux Falls
605-335-0780	T	Sioux Falls

Tennesee (TN)

615-968-1130	G	Bristol
615-756-1161	G	Chattanooga
615-265-1020	T	Chattanooga
901-424-2114	T	Jackson
615-673-8901	C	Knoxville
615-523-5500	G	Knoxville
615-690-1543	T	Knoxville
901-452-8530	C	Memphis
901-452-1710	C	Memphis
901-521-0215	G	Memphis
901-527-8006	T	Memphis
615-366-1947	C	Nashville
615-244-3702	G	Nashville
615-885-3530	T	Nashville
615-482-9080	T	Oak Ridge

Texas (TX)

915-676-9151	G	Abilene
915-672-4611	T	Abilene
806-372-6934	G	Amarillo
806-383-0304	T	Amarillo
512-444-7234	C	Austin
512-928-1130	G	Austin
512-444-3280	T	Austin
713-422-9746	T	Baytown
512-541-2251	T	Brownsville
409-779-0184	T	Bryan
409-779-0184	T	College Station
512-884-9030	G	Corpus Christi
512-883-8050	T	Corpus Christi
214-761-0599	C	Dallas
214-761-9040	C	Dallas
214-748-0127	G	Dallas
214-638-8888	T	Dallas
817-565-9273	T	Denton
915-565-4661	C	El Paso
915-565-4670	C	El Paso
915-532-7907	G	El Paso
915-533-1453	T	El Paso
817-870-2461	C	Ft. Worth
817-870-2468	C	Ft. Worth
817-332-4307	G	Ft. Worth
817-877-3630	T	Ft. Worth
409-762-4382	G	Galveston
409-765-7338	T	Galveston
713-225-2550	C	Houston
713-225-2330	C	Houston
713-227-1018	G	Houston
713-556-6700	T	Houston
817-634-2810	T	Killeen
512-225-8004	G	Lackland
214-236-3196	G	Longview
214-236-4041	T	Longview
806-763-5081	C	Lubbock
806-747-4121	G	Lubbock
806-797-0765	T	Lubbock
512-631-0020	T	McAllen
915-687-1464	C	Midland
915-561-9811	G	Midland
915-683-5645	T	Midland
409-722-3720	G	Nederland
409-724-0726	T	Nederland
915-561-9811	G	Odessa
915-563-3745	T	Odessa
915-944-7621	G	San Angelo
512-435-3883	C	San Antonio
512-225-8004	G	San Antonio
512-225-8002	T	San Antonio
915-561-9811	G	Terminal
409-765-7338	T	Texas City
214-592-1372	T	Tyler
817-752-9743	G	Waco
817-752-1642	T	Waco
817-761-1315	T	Wichita Falls

Utah (UT)

801-627-1630	G	Ogden
801-627-2022	T	Ogden
801-375-0645	T	Provo
801-521-2890	C	Salt Lake City
801-359-0149	G	Salt Lake City
801-364-0780	T	Salt Lake City

Virginia (VA)

202-429-7896	G	Alexandria
202-429-7896	G	Annandale
703-841-9834	C	Arlington
703-691-8200	T	Arlington
804-973-8815	C	Charlottesville
804-971-1505	G	Charlottesville
804-971-1001	T	Charlottesville
804-625-1186	G	Chesapeake
703-352-7500	C	Fairfax
202-429-7896	G	Fairfax
703-691-8390	T	Fairfax
703-691-8200	T	Fairfax
202-429-7896	G	Falls Church
804-245-0021	C	Hampton
703-435-1800	G	Herndon
804-528-1903	T	Lynchburg
804-744-4860	T	Midlothian
804-596-6600	G	Newport News
804-596-7608	T	Newport News
804-461-6128	C	Norfolk
804-461-6167	C	Norfolk
804-625-1186	G	Norfolk
804-855-7751	T	Norfolk
804-862-4700	T	Petersburg
804-625-1186	G	Portsmouth
804-855-7751	T	Portsmouth
804-358-8274	C	Richmond
804-788-9902	G	Richmond
804-744-4860	T	Richmond
703-344-2036	G	Roanoke
703-344-2762	T	Roanoke
202-429-7896	G	Springfield
202-429-7896	G	Vienna
804-625-1186	G	Virginia Beach
804-872-9592	T	Williamsburg

Vermont (VT)

802-864-0808	G	Burlington
802-658-2123	T	Burlington
802-229-4966	G	Montpelier
802-223-3519	T	Montpelier

Washington (WA)

206-939-9982	G	Auburn
206-447-9012	G	Bellevue
206-647-0666	T	Bellingham
206-825-7720	T	Enumclaw
206-577-5835	G	Longview
206-754-0460	G	Olympia
206-438-2772	T	Olympia
509-375-3367	T	Richland
206-241-9111	C	Seattle
206-241-7023	C	Seattle
206-447-9012	G	Seattle
206-285-0109	T	Seattle
509-326-0515	C	Spokane
509-455-4071	G	Spokane
509-747-4105	T	Spokane
206-627-1791	G	Tacoma
206-272-1503	T	Tacoma
206-693-0371	T	Vancouver
509-663-6227	G	Wenatchee
509-453-1591	T	Yakima

Wisconsin (WI)

414-722-5580	T	Appleton
608-365-6883	T	Beloit
414-475-6381	C	Brookfield
414-475-6935	C	Brookfield
414-785-1614	T	Brookfield
715-832-1211	G	Eau Claire
715-832-1354	T	Eau Claire
414-432-2815	G	Green Bay
414-432-3064	T	Green Bay
608-785-1450	T	La Crosse
608-256-6525	C	Madison
608-257-5010	G	Madison
608-221-4211	T	Madison
608-221-0891	T	Madison
414-475-6935	C	Milwaukee
414-475-6381	C	Milwaukee
414-271-3914	G	Milwaukee
414-785-1614	T	Milwaukee
414-722-5580	T	Neenah
414-235-1082	T	Oshkosh
414-552-7217	G	Racine
414-632-3006	T	Racine
414-334-1240	T	West Bend

West Virginia (WV)

304-768-9700	C	Charleston
304-345-6471	G	Charleston
304-345-9575	T	Charleston
304-736-2331	C	Huntington
304-523-2802	G	Huntington
304-525-4406	T	Huntington
304-292-2175	T	Morgantown
304-295-9371	C	Parkersburg
304-428-8511	T	Parkersburg
304-232-3589	C	Wheeling

Wyoming (WY)

307-265-5167	G	Casper
307-235-0164	T	Casper
307-638-4421	G	Cheyenne

CompuServe CIS Commodore Information Service

My local CompuServe Number is:

____

My CompuServe Account Number is:

____

T – TOP
TOP menu page. Goes directly page CIS-1

M – MENU
Previous MENU. Goes back to the menu page that points to the current page. A single <Enter> will also return to the last menu if there isn't a next page.

G – GO
Go n. . . Go directly to page 'n'. 'n' may either be an information provider/number combination, like TRS–1, or a number alone. The latter will refer to the current information provider.

H – HELP
Displays HELP file.

S – SCROLL
S n. . . SCROLL from item 'n'. (Note: there MUST be a space between S and the page number. Example: S 4 will output pages until the last page in a series is reached. If at a menu page, 'n' specifies the menu item to scroll from.

OFF or BYE
These commands will disconnect you from the Information Service immediately.

F – FORWARD
FORWARD a page. Displays the next page in a series of pages. A single <Enter> key will do the same thing.

B – BACKWARD
Returns to the preceding page.

P – PREVIOUS
Go to the PREVIOUS item from last selected menu. If 5 was the last choice, P will display item 4.

N – NEXT
Go to the NEXT item from last selected menu. If 5 was the last choice, NEXT will display item 6.

R – RESEND
RESEND the current page. This is useful if the current page has scrolled off the screen or after a HELP command.

Control Characters

Control characters are entered by holding down the Control key while at the same time pressing the character key. Some keyboards do not have a CONTROL key. Programmers usually designate the OFF/RVS key as the Control key. But it is not a true Control key. Therefore the RVS key is (most often) pressed and released before entering the character.

The control characters most often used are: **↑ = Control**

- **↑C** interrupts display or a program's execution so that you can enter another menu selection or command.
- **↑U** deletes the line which you are currently typing.
- **↑V** redisplays the partial line you are typing and allows you to continue typing the line.
- **↑H** backspaces, deleting the character that was there. Note that the character may not disappear from your screen but it is no longer recognized by the computer.
- **↑A** temporarily suspends output at the end of the current line. Enter ↑Q to resume.
- **↑S** temporarily suspends output immediately, even if it is in the middle of a line. Enter ↑Q to resume output.
- **↑Q** resumes output after ↑A or ↑S.
- **↑O** stops output which is in process (cannot be resumed).
- **↑P** interrupts output and takes you to a command prompt.

CompuServe Category Index

SIG = Special Interest Group

Category	Page
AAMSI Communications	AAM
AAMSI SIG	SFP-5
AOPA Forum	AOP
AP Datastream	SPD-1005
AP Videotex, Business	APV
AP Videotex, Entertainment	APV
AP Videotex, Politics	APV
AP Videotex, Weather	APV
AP Videotex, World News	APV
ASCMD SIG	SFP-7
ASI Flight Operations	ASI-11
ASI Monitor	ASI-10
ASI Service Difficulty	ASI-12
Academic Amer. Encyclopedia	AAE
Access Phone Numbers	LOG-50
Adventure	GAM-8
Aircraft Insurance	AVL
Alternative Educ. Services	AES
Altertext Report	ALT
American Ski Association	SKI
Apple User Group SIG	PCS-51
Arcade SIG	HOM-138
Astrology	GAM-45
Atari SIG	PCS-132
Athlete's Outfitter	HAN
Aunt Nettie	NET
AutoNet	ATO
Aviation Rules & Reg.	AVR
Aviation SIG (AVSIG	SFP-6
Aviation Safety Institute	ASI
Aviation Weather	AWX
Bacchus Data Services	VIN
Backgammon	GAM-31
Banking Services	HOM-45
Banshi	GAM-30
Belmont Golf Association	BEL
Biorhythms	GAM-29
Blackjack	GAM-60
Bridge	GAM-18
Bulletin Board	HOM-30
Business & Law Review	BLR
CB	CB-10
CB Interest Group SIG	HOM-9
CB Society	CUP
CEMSIG SIG	CEM-5
CP Business Info Wire	BIW
CP/M Users Group SIG	PCS-47
Calculate A Raise	HOM-15
Calculate Net Worth	HOM-16
Changing Password	CIS-175
Changing Terminal Type	CIS-9
Checkbook balancer	HOM-14
Children's Games	TMC-27
Civil War	GAM-14
Clarke School for the Deaf	CSD
CoalScoop	CMP
College Press Service	CPS
Color Computer SIG	PCS-126
Color Graphics	CIS-91
Columbus Chamber of Commerce	CCC
Command Summary	CIS-58
Commodore	CBM
Commodore 64 SIG	PCS-156
Commodore Programming Sig	PCS-116
Commodore VIC20 & Pet/CBM	PCS-155
Communication Industry	SFP-35
Comp-U-Store	CUS
CompuServe Rates	BIL
CompuServe logon instruct	LOG
CompuServe's Softex	PCS-40
Computer Art SIG	PCS-157
Computer Job Bank	TDC-4
Computer Resume Bank	TDC-4
Computer Wire, The	TDC-4
Computing Across America	CAA
Computing Tutorials	PCS-121
Concentration	GAM-32
Cook's Underground	HOM-109
DISCOVER ORLANDO	ORL
DataPac logon instruct	LOG-41
Democratic Forum	HOM-39
Department of State	DOS
Dice	GAM-33
Digital Research Inc.	DRI
Direct Connection, The	TDC
EMAIL	EMA
EMI Flight Planning	EMI
Economic News	FIN-10
Educational Research Sig	HOM-28
Educators' SIG	HOM-137
Edutech	CAI
Election '84	VOT
Electronic Bounce Back	EBB
Electronic Gourmet	HMS

Category	Page
Entertainment SIG	HOM-29
Environmental SIG	SFP-38
EpsOnLine	PCS-19
Evans Economic Inc.	EEI
FOI Newsline - FDA Info.	FOI
Family Matters SIG	HOM-144
Fantasy	GAM-16
FasterMind	GAM-17
Fedwatch Newsletter	MMS
Feedback to CompuServe	CIS-8
Fifth Avenue Shopper	FTH
Financial Forecasts	FIN-4
Financial Services	FIN-20
Fire Fighters' SIG	SFP-36
Firstworld Travel Club	TVL
Food Buyline SIG	HOM-151
Football	GAM-27
Fur trader	GAM-36
GameSIG Archives	GSA
Gandolf's Reports	GAN
Golf	GAM-21
Golf SIG	HOM-129
Gomoku	GAM-22
Good Earth SIG	HOM-145
Government Publications	GPO
HamNet SIG	HOM-11
Hammurabi	GAM-37
Handicapped Users' Database	HUD
Hangman	GAM-23
Heath User Group SIG	PCS-48
Heathkit Catalog	HTH
Hi-Tech Forum SIG	CCC-150
Hollywood Hotline	HHL
Home Management	HOM-80
Howard Sams' Books	SAM
Human Sexuality	HSX
Huntington National Bank	HNB
IBM-PC SIG	PCS-131
Incorporating Guide	INC
Index	IND
Industry Standard Databases	TDC-4
InfoText	IFT
InfoWorld	INF
Information on Demand	IOD
Intelligence Test	TMC-32
Internal Revenue Services	IRS
Kaypro Users Forum	PCS-25
Kesmai	GAM-46
LSI SIG	PCS-49
Legal SIG	SFP-40
Literary SIG	HOM-136
Loan Amortization	HOM-17
Lunar Lander	GAM-24
MNET80 SIG	PCS-54
MUSUS SIG	PCS-55
Magic Cube Solution	GAM-35
Max Ule's Tickerscreen	TKR
Maze	GAM-38
MegaWars I	GAM-20
MegaWars II	GAM-55
MegaWars III	GAM-15
MicroQuote	FIN-9
MicroShoppe	MCS
Microsoft SIG	PCS-145
Military Vets Forum	SFP-10
Mine-Equip	MIN-100
Miner's Underground	SFP-44
Money Market Services	MMS
Monthly Charges	MON
Mugwump	GAM-39
Multi-Player GameSIG	GAM-300
Music Information Service	MUS
Music SIG	HOM-150
NOAA Weather Wire	WEA
NWS Aviation Weather	AWX
Narrow-Gage Scout	LMC
National Issues SIG	HOM-132
National Water Well Assoc	WWA
Netwits Database	WIT
Netwits SIG	WIT-100
New Adventure	GAM-59
News-A-Tron	NAT
Newspapers	HOM-10
Node Abbreviations	LOG-51
OS9 SIG	PCS-18
Official Airline Guide	OAG
Ohio Scientific SIG	PCS-125
Orch-90 SIG	HOM-13
Outdoor SIG	HOM-38
PDP-11	PCS-53
PGA Official Tour Guide	PGA
PR and Marketing Forum	SFP-48
Pan Am Travel Guide	PAN
Panasonic SIG	PCS-114

Category	Page
Parenting & Family Life	PFL
Pascal SIG	PCS-55
Peak Delay Guide	PDG
Personal Computing	PCS
Personal File Area	CIS-174
Personality Profile	TMC-17
Popular Science, Autos	PSC
Popular Science, Energy	PSE
Popular Science, New Product	PSP
PowerSoft's XTRA-80	PCS-56
Primetime Radio Classics	PRC
Product Ordering	CIS-54
Programmer's SIG	PCS-158
Quick Quote	FIN-20
Quick Reference List	QUICK
RCA SIG	PCS-57
Rapaport Diamond Broker	RDC
Religion SIG	HOM-33
Republican Forum	HOM-41
Reversi	GAM-40
Rick's Arcade Center	ARC
Roulette	GAM-42
SAVINGS-SCAN	SAV
SHO-TIME Movie Catalog	MOV
Scott Adams' Games	GAM-28
Scramble	GAM-43
SeaWar	GAM-57
Shareholders Freebies	FRE
Shawmut Bank of Boston	SHW
Shop-at-home	HOM-40
Ski SIG	HOM-36
Social Security Administration	SSA
Society of Mining Engineers	SME
Software Author's SIG	PCS-117
Space SIG	HOM-127
Space Trek	GAM-26
Space War	GAM-25
Sports SIG	HOM-110
StL Post-Dispatch, Autos	SPD
StL Post-Dispatch, Business	SPD
StL Post-Dispatch, Jobs	SPD
StL Post-Dispatch, Real Est.	SPD
StL Post-Dispatch, Sports	SPD
StL Post-Dispatch, U.S. News	SPD
StL Post-Dispatch, Classified	SPD
Standard & Poor's	FIN-20
State Capital Quiz	TMC-44
Stevens Business Reports	SBR
TRS-80 Professional Forum	PCS-21
TRS80 Model 100 SIG	PCS-154
TYMNET logon instructions	LOG-11
Tandy Newsletter	TRS
TeleComm SIG	PCS-52
Telenet logon instruct	LOG-20
Terminal Software	PCS-20
Texas Instruments Forum	PCS-27
Text Editors	PCS-20
The Business Wire	TBW
The College Board	TCB
The Electronic Mall	EM
The Multiple Choice	TMC
The National Satirist	KCS
The New Tech Times	NTT
The World of Lotus	LOTUS
Touch-Type Tutor	TMC
Travel Fax	ESC
Travel SIG	HOM-157
TravelVision	TRV
Trivia Test	TMC
Unified Management	UMC
United American Bank	HOM-152
User Directory	HOM-4
VAX SIG	PCS-16
VIDTEX Information	VID
Value Line Financials	FIN-20
Value Line Projections	FIN-18
Veterinarians Forum	SFP-37
Victory Garden	VIC
Video Information	VIF
Washington Post, Business	TWP-12
Washington Post, Editorials	TWP
Washington Post, Financial	TWP
Washington Post, Gov't News	TWP
Washington Post, Politics	TWP-15
Washington Post, Sports	TWP
Washington Post, U.S. News	TWP
Washington Post, World News	TWP
West Coast Travel	WCT
What's New	NEW
Whole Earth Software SIG	WEC
Words of Wit & Wisdom	WWW
Work-at-home SIG	HOM-146
Worldwide Exchange	WWX
Wumpus	GAM-44

Bulletin Boards By Area Code

24h Denotes 24-hour operation ↔ Multi-User System $ Pay System, Password Required ⚤ Sexually Oriented BBS
● Nighttime Operation ★ 1200 Baud Allowed Ⓟ Password Required † Religious orientation

201		
☐ 201-864-5345	ABBS Apple-Mate, New York, NY	
☐ 201-835-7228	ABBS CCNJ, Pompton Plains, NJ	
☐ 201-891-7441	A.C.C.E.S.S., Wyckoff, NJ	24h
☐ 201-790-5910	Aphrodite-E, Haledon, NJ	⚤
☐ 201-627-5151	Conference-Tree Flagship, Rockaway, NJ	24h
☐ 201-272-3686	Dial-Your-Match #14, Cranford, NJ	⚤
☐ 201-462-0435	Dial-Your-Match #21, Freehold, NJ	⚤
☐ 201-486-2956	Forum-80, Linden, NJ	24h
☐ 201-528-6623	Forum-80 Monmouth, Brielle, NJ	24h
☐ 201-994-9620	Net-Works The Barn, Livingston, NJ	24h
☐ 201-736-4630	Pirates Distributing	
☐ 201-366-2209	Pirates I/O	
☐ 201-423-0810	Places Unknown	
☐ 201-790-6795	Photo-80, Haledon, NJ	
☐ 201-932-3887	PMS Rutgers Univ. Microlab, Piscataway, NJ	
☐ 201-887-8874	RATS System, Whippany, NJ	
☐ 201-584-9227	RCP/M Flanders, NJ	24h ★
☐ 201-272-1874	RCP/M RBBS Cranford, NJ	24h
☐ 201-775-8705	RCP/M RBBS Ocean, NJ	★
☐ 201-747-7301	RCP/M RBBS Paul Bogdanovich, NJ	
☐ 201-932-3879	RCP/M RBBS Rutgers, New Brunswick, NJ	24h
☐ 201-625-1797	RCP/M The C-Line, NJ	●
☐ 201-233-5997	Sherwood Forest	

202		
☐ 202-364-8617	Aladdin's Lamp	
☐ 202-276-8342	ARMUDIC Washington, DC	
☐ 202-363-8165	NWDS	
☐ 202-337-4694	Program Store of DC, Washington, DC	24h
☐ 202-678-9947	Ware-House III	

203		
☐ 203-744-4644	Bullet-80, Danbury, CT	
☐ 203-888-7952	Bullet-80, Seymour, CT	
☐ 203-834-0026	Spectre-80	
☐ 203-746-5763	Telcom 7, New Fairfield, CT	24h

204		
☐ 204-785-8742	Selkirk BBS, Selkirk, MB, CAN	24h

205		
☐ 205-492-0373	Bullet-80, Gadsden, AL	24h
☐ 205-272-5069	Forum-80, Montgomery, AL	
☐ 205-972-1685	Pentagon	
☐ 205-895-6749	RCP/M RBBS NACS/UAH, Huntsville, AL	24h

206		
☐ 206-935-9119	ABBS Apple Crate I, Seattle, WA	
☐ 206-244-5438	ABBS Apple Crate II, Seattle, WA	
☐ 206-866-9043	A.C.C.E.S.S., Olympia, WA	24h
☐ 206-621-8665	Anchor CP/M	
☐ 206-525-5410	Apple Crate I, Seattle, WA	
☐ 206-546-6239	ARBB, Seattle, WA	
☐ 206-524-0203	Call-A.P.P.L.E, Seattle, WA	
☐ 206-256-6624	Dial-Your-Match #16, Seattle, WA	⚤
☐ 206-723-3282	Forum-80, Seattle, WA	
☐ 206-883-0403	JCTS Redmond, WA	24h
☐ 206-767-7777	Kingdom of Seven, Seattle, WA	
☐ 206-527-0897	Mail Board-82, Seattle, WA	24h
☐ 206-762-5141	Mini-Bin, Seattle, WA	24h
☐ 206-334-7394	MSG-80 Everett, WA	
☐ 206-743-6021	NWWCUG Edmunds, Seattle, WA	
☐ 206-783-9798	Pirates of Puget Sound, Seattle, WA	
☐ 206-486-2368	PMS Software Unlimited, Kenmore, WA	24h
☐ 206-357-7400	RCP/M Olympia, WA	24h
☐ 206-458-3086	RCP/M RBBS Yelm, Olympia, WA	
☐ 206-763-8879	Seacomm-80, Seattle, WA	24h

207		
☐ 207-839-2337	RCP/M Programmers Anonymous, Gorham, ME	24h ★

209		
☐ 209-298-1328	Dial-Your-Match #26, Clovis, CA	⚤

212		
☐ 212-896-0519	(?) Queens, NY	
☐ 212-933-9459	Bronx BBS, New York, NY	
☐ 212-740-5680	Bullet-80, New York, NY	24h
☐ 212-897-3392	Comm-80, Queens, NY	24h
☐ 212-991-1664	Connection-80, Manhattan, NY	
☐ 212-441-3755	Connection-80, Woodhaven, NY	24h
☐ 212-631-1788	Kracker's Kastle	
☐ 212-541-5975	MMMMM#2, New York, NY	⚤
☐ 212-410-0949	Net-Works, Brooklyn, NY	
☐ 212-626-0375	Nybbles-80, NY	
☐ 212-997-2488	PMS Mcgraw-Hill Books, New York, NY	
☐ 212-255-7240	RCP/M RBBS Manhattan, New York, NY	24h ★
☐ 212-442-3874	Sister, Staten Island, NY	24h
☐ 212-799-4649	TCBBS Astrocom, New York, NY	24h
☐ 212-362-1040	TCBBS B.A.M.S. New York, NY	24h

213		
☐ 213-829-1140	ABBS Computer Conspiracy, Santa Monica, CA	
☐ 213-459-6400	ABBS Pacific Palisades, Los Angeles, CA	
☐ 213-537-3378	Access One, CA	
☐ 213-564-7636	All Night BBS, CA	
☐ 213-991-1604	Alpha Byte, CA	
☐ 213-851-0780	Aware II, Los Angeles, CA	
☐ 213-394-5950	BBS B.R., Los Angeles, CA	24h
☐ 213-649-1489	BBS IBM PC, Culver City, CA	24h ★
☐ 213-930-2578	CIA	
☐ 213-657-1799	Computer Connection, Los Angeles, CA	
☐ 213-372-4800	Conference-Tree Kelp Bed, Los Angeles, CA	
☐ 213-394-1505	Conference-Tree, Santa Monica, CA	
☐ 213-633-5463	Data-Mate, Canoga Park, CA	⚤
☐ 213-346-1849	Dec-Line, Woodland Hills, CA	24h ↔
☐ 213-842-3322	Dial-Your-Match #1, CA	⚤
☐ 213-990-6830	Dial-Your-Match #22, CA	⚤
☐ 213-783-2305	Dial-Your-Match #4, CA	⚤
☐ 213-345-1047	Dial-Your-Match #9, CA	⚤
☐ 213-347-9780	Dr. Falcon's Retreat, Canoga Park, CA	★
☐ 213-428-5206	Dragon's Game System	Ⓟ = dragon
☐ 213-789-9512	Electric Line Connection, Sherman Oaks, CA	
☐ 213-840-8066	Fantasy Plaza	
☐ 213-287-1363	Greene Machine, Temple City, CA	
☐ 213-445-3591	Greene Machine, Fricaseed Chicken, Arcadia, CA 24h	
☐ 213-431-1443	Greene Machine, Los Alamitos, CA	
☐ 213-591-7239	Groundstar System, Long Beach, CA	24h
☐ 213-366-1238	HBBS Mog-ur, Granada Hills, CA	24h ★
☐ 213-477-4605	Interface, Los Angeles, CA	
☐ 213-947-8128	Kluge Computer	24h ★
☐ 213-631-3186	L.A. Interchange, Los Angeles, CA	24h
☐ 213-478-5478	Master World, Los Angeles, CA	
☐ 213-470-5912	Mad Board From Mars, Los Angeles, CA	
☐ 213-390-3239	MMMMM#1, Santa Monica, CA (line One)	★ ⚤
☐ 213-450-4580	MMMMM#1, Santa Monica, CA (line Two)	⚤
☐ 213-452-6111	MMMMM#3, Marina Del Rey, CA	⚤
☐ 213-821-2257	MMMMM#4, Lawndale, CA	⚤
☐ 213-336-5535	Net-Works Coin Games, Los Angeles, CA	
☐ 213-859-0894	Net-Works Computer World, Los Angeles, CA	24h
☐ 213-345-3670	Net-Works Encino, CA	
☐ 213-388-5198	Net-Works Magnetic Fantasies, Los Angeles, CA	
☐ 213-454-3075	Net-Works Pirate's Inn, CA	
☐ 213-473-2754	Net-Works Softworx, West Los Angeles, CA	
☐ 213-881-6880	Novation Co., Los Angeles, CA	Ⓟ = cat
☐ 213-980-5643	Oracle, North Hollywood, CA	⚤
☐ 213-784-0204	Outer Limits # 1, Van Nuys, CA	24h
☐ 213-782-8390	Outer Limits # 2, Van Nuys, CA	
☐ 213-360-0211	Phantoms Hollow, Granada Hills, CA	
☐ 213-472-4287	Pirates Mountain, Los Angeles, CA	
☐ 213-395-9813	Pirates Paper, Santa Monica, CA	
☐ 213-331-3574	PMS, Los Angeles, CA	24h
☐ 213-368-5801	RBBS, San Fernando, CA	
☐ 213-395-0460	RBBS, Santa Monica, CA	★
☐ 213-799-1632	RCP/M CBBS, Pasadena, CA	24h
☐ 213-360-5053	RCP/M, Granada Hills, CA	24h
☐ 213-296-5927	RCP/M, Los Angeles, CA	24h
☐ 213-541-2503	RCP/M RBBS GFRN Data Exchange Palos Verdes, CA	24h ★
☐ 213-653-6398	RCP/M RBBS, Hollywood, CA	24h
☐ 213-973-2374	RCP/M RBBS IBM PC, Hawthorne, CA	★
☐ 213-577-9947	RCP/M RBBS, Pasadena, CA	24h ★
☐ 213-447-0681	The Frigate	
☐ 213-375-6137	Torture Chamber, Los Angeles, CA	
☐ 213-357-2038	Twilight Zone	
☐ 213-859-2735	Ye Pawn Shoppe, Los Angeles, CA	

214		
☐ 214-424-3862	ABBS Dallas Info Board, Dallas, TX	
☐ 214-960-7654	ABBS Teledunjon III, Dallas, TX	
☐ 214-631-7747	ABBS The Pulse, Dallas, TX	24h ⚤
☐ 214-289-1386	BBS-80 Daltrug, Dallas, TX	24h
☐ 214-644-4781	Net-Works Apple Shack, TX	
☐ 214-361-1386	Net-Works Dallas, TX	
☐ 214-239-5842	Net-Works Eclectic Computer Systems, Dallas, TX	
☐ 214-824-7455	Net-Works Winesap, TX	
☐ 214-931-8274	RCP/M CBBS, Dallas, TX	●
☐ 214-241-1939	RCP/M CBBS Maxicom, Farmers Branch, TX	24h ★
☐ 214-247-5307	RCP/M CBBS Maxicom, Line 2	
☐ 214-769-3036	TBBS Hawkins, TX	24h ★

215		
☐ 215-364-2180	Bullet-80, Langhorne, PA	
☐ 215-855-3809	Comnet-80, North Wales, PA	
☐ 215-563-9815	Datanet 1200 Baud	
☐ 215-563-9211	Datanet 300 Baud	
☐ 215-434-3998	Hermes-80, Allentown, PA	
☐ 215-435-3388	Lehigh Press BBS, Allentown, PA	
☐ 215-244-0864	Net-Works Galaxy One, PA	
☐ 215-398-3937	RCP/M RBBS, Allentown, PA	24h
☐ 215-446-7670	Video Ace	
☐ 215-363-0563	Video Fantasies, Langhorne, PA	

216		
☐ 216-745-7855	ABBS Akron Digital Group, Akron, OH	24h
☐ 216-757-3711	BBS Computer Applications Co., Poland, OH	
☐ 216-729-2769	Bullet-80, Chesterland, OH	
☐ 216-645-0827	Comnet-80, Akron, OH	24h ★
☐ 216-486-4176	Forum-80, Cleveland, OH	★
☐ 216-845-3179	Genius' Modemline	
☐ 216-724-2125	Infoex-80, Akron, OH	24h
☐ 216-875-4582	Micro-Com, Louisville, OH	24h
☐ 216-832-8392	PMS Massillon, OH	24h
☐ 216-867-7463	PMS Raug, Akron, OH	24h

217		
☐ 217-529-1113	Bullet-80, Springfield, IL	
☐ 217-877-1544	Hacker's Haven	
☐ 217-753-4309	MCMS Word Exchange, Springfield, IL	24h
☐ 217-429-4738	Net-Works C.A.M.S., Decatur, IL	24h

☐ 217-429-6310	Rag Time Phreak, Decatur, IL	
☐ 217-875-5579	South Pole	

218

☐ 218-727-2184	Northeast Minnesota Net, MN	

301

☐ 301-267-7666	A.C.C.E.S.S., Annapolis, MD	24h
☐ 301-730-0922	ABBS Computer Crossroads, Columbia, MD	
☐ 301-881-0846	Alcatraz	
☐ 301-587-2132	ARMUDIC Computer Age, Baltimore, MD	
☐ 301-984-3772	ASCII	
☐ 301-937-4339	BBS IBM PC, Beltsville, MD	24h
☐ 301-460-0538	BBS IBM PC, Bethesda, MD	24h
☐ 301-251-6293	BBS IBM PC, Gaithersburg, MD	24h
☐ 301-949-8848	BBS IBM PC, Rockville, MD	24h
☐ 301-948-5717	CBBS CPEUG/ICST, Gaithersburg, MD	
☐ 301-640-0498	Centaur Island	
☐ 301-543-9429	Net-Works Computer Island, MD	
☐ 301-840-8588	Connection-80, Gaithersburg, MD	24h
☐ 301-926-3470	Doctor's Office	
☐ 301-593-7033	Handicapped Exchange, Silver Spring, MD	24h
☐ 301-560-9555	Micro Encounter	
☐ 301-983-8293	Mission Control	
☐ 301-953-3341	Net-Works Comm Center NW3NAGAD, Laurel, MD	
☐ 301-869-8747	Pirates Landing	
☐ 301-764-1995	PMS, Baltimore, MD	24h
☐ 301-465-3176	PMS, Ellicott City, MD	
☐ 301-653-3413	PMS, Pikesville, MD	
☐ 301-356-5895	Possesion	
☐ 301-994-0399	Program Store BBS Baltimore, MD	24h
☐ 301-229-3196	RCP/M RBBS, Bethesda, MD	
☐ 301-661-2175	RCP/M RBBS BHEC, Baltimore, MD	24h
☐ 301-953-3753	RCP/M RBBS, Laurel, MD	24h
☐ 301-344-9156	Remote Northstar Nasa, Greenbelt, MD	
☐ 301-565-9051	Tech-Link, Forest Glen, MD	24h

303

☐ 303-759-2625	ABBS, Denver, CO	
☐ 303-333-1132	American BBS	
☐ 303-698-7620	Chess Board, Denver, CO	
☐ 303-753-1554	Cheyenne Mountain, Denver, CO	
☐ 303-690-4566	Connection-80, Denver, CO	24h
☐ 303-465-2027	Forbidden Zone	
☐ 303-399-8858	Forum-80 #2, Denver, CO	24h
☐ 303-693-1064	GBBSII, Denver, CO	●
☐ 303-469-7541	GBBSII Apple Pi, CO	24h
☐ 303-343-8401	GBBSII Aurora-Net, Denver, CO	24h
☐ 303-750-3783	GBBSII Eamon, Denver, CO	● ★
☐ 303-443-3367	GBBSII Off The Wall, Denver, CO	24h
☐ 303-423-3156	Laboratory I	
☐ 303-751-2063	Laboratory II, Denver, CO	
☐ 303-694-2871	Magic Window, Denver, CO	
☐ 303-986-5039	Mansion, Denver, CO	
☐ 303-985-9184	Neutral Zone, Denver, CO	
☐ 303-499-9169	RCP/M Boulder, CO	●
☐ 303-781-4937	RCP/M Cug-Note, Denver, CO	24h
☐ 303-634-1158	RCP/M RBBS Arvada Elect, Colorado Springs, CO 24h	
☐ 303-985-1108	RCP/M RBBS Lakewood, Denver, CO	24h
☐ 303-598-3995	RCP/M RBBS Pinecliffe, CO	24h ★
☐ 303-444-7231	Remote Northstar, Denver, CO	
☐ 303-279-5657	Robotics-BBS	
☐ 303-427-7114	Testing Zone	
☐ 303-796-8708	U called it U name it	

304

☐ 304-925-3338	Century 21st	
☐ 304-345-8280	Net-Works Charleston, WV	
☐ 304-744-2253	Pirate-80	
☐ 304-372-4486	The Morg	

305

☐ 305-486-2983	ABBS Byte Shop, Ft. Lauderdale, FL	
☐ 305-261-3639	ABBS Byte Shop, Miami, FL	
☐ 305-848-3802	ABBS, West Palm Beach, FL	
☐ 305-238-1231	AMIS Apogee, Miami, FL	
☐ 305-246-1111	BBS Homestead, FL	
☐ 305-392-5927	Boca Harbor	
☐ 305-432-5969	Cable Box	
☐ 305-969-0000	Color Dimension 300, West Palm Beach, FL	
☐ 305-644-8327	Connection-80, Orlando, FL	24h
☐ 305-894-1886	Connection-80, Winter Garden, FL	24h
☐ 305-391-3893	C.O.P.S	
☐ 305-772-4444	Forum-80 Ft. Lauderdale, FL	24h
☐ 305-965-4388	Greene Machine, West Palm Beach, FL	⚤
☐ 305-968-8653	Greene Machine Corsair, West Palm Beach, FL	
☐ 305-683-6044	Infoex-80, West Palm Beach, FL	24h
☐ 305-686-3695	Micro-80, West Palm Beach, FL	
☐ 305-755-5560	Mordor	
☐ 305-772-1076	Net-Works Apple Barrel, FL	
☐ 305-948-8000	Net-Works Big Apple, Miami, FL	
☐ 305-686-4862	Notebook, West Palm Beach, FL	
☐ 305-427-6300	Personal Msg. System-80, Deerfield Beach, FL	24h ★
☐ 305-335-8640	Pirates Loft II	
☐ 305-854-6398	Pirates Reef	
☐ 305-823-2756	Pirates Reef II	
☐ 305-763-1654	Project Blue Book	
☐ 305-830-4340	RCP/M RBBS IBM PC, Orlando, FL	24h ★
☐ 305-671-2330	RCP/M RBBS, Orlando, FL	24h ★
☐ 305-645-5543	TBBS Pizza-Net, Orlando, FL	24h
☐ 305-798-1615	Temple Toa-Rin	
☐ 305-393-7122	The Freezer	
☐ 305-525-1192	Trade-80, Ft. Lauderdale, FL	

307

☐ 307-637-6045	PET BBS SE Wyoming PUG	24h

309

☐ 309-692-6502	ABBS Peoria, IL	
☐ 309-797-8535	Mystery Castle	
☐ 309-342-7178	Net-Works Magie, Galesburg, IL	
☐ 309-729-9518	Phantom's Mansion	
☐ 309-944-5455	RCP/M Geneseo, IL	

312

☐ 312-882-2926	ABBS Code, Glen Ellyn, IL	24h
☐ 312-475-4884	ABBS Gamemaster, Chicago, IL	24h
☐ 312-973-2227	ABBS Rogers Park, Chicago, IL	
☐ 312-475-5282	ABBS Video Adv. Movie Marquee, Evanston, IL	
☐ 312-392-2403	ACS Arlington Heights, IL	
☐ 312-445-1130	ACS Chicago, IL	
☐ 312-789-3610	AMIS, Clarendon Hills, IL	24h
☐ 312-674-2578	AT&T Phone Center	
☐ 312-991-8887	BBS IBM PC, Niles, IL	24h
☐ 312-882-4227	BBS IBM PCmodem, Chicago, IL	24h ★
☐ 312-376-7598	BBS IBM PCmodem, Chicago, IL	24h
☐ 312-598-4861	Cass-80, Hickory Hills, IL	
☐ 312-897-9037	CBBS Aurora Computer Peripherals, Aurora, CO. 24h	
☐ 312-545-8086	CBBS Chicago, IL	24h
☐ 312-259-8086	CBBS Ward And Randy's, Chicago, IL	
☐ 312-957-3924	C.M.M.S., Chicago, IL	24h
☐ 312-674-6502	Commodore Video King, IL	
☐ 312-243-1046	Dial-Your-Match #39, Chicago, IL	⚤
☐ 312-622-4442	Greene Machine, Chicago, IL	⚤
☐ 312-296-3883	Interface BBS (Atari), Chicago, IL	
☐ 312-674-9246	Marvin	
☐ 312-927-1020	MCMS C.A.M.S., Chicago, IL	24h ★
☐ 312-260-0640	MCMS Metro West Database, Chicago, IL	24h ★
☐ 312-462-7560	MCMS P.C.M.S., Wheaton, IL	24h ★
☐ 312-351-4374	MCMS Waco Hot Line, Schaumburg, IL	24h (private)
☐ 312-279-4399	Midwest Pirate System	
☐ 312-759-9191	Mother	
☐ 312-295-7284	Net-Works Adventure's Inn, Lake Forest, IL	24h
☐ 312-685-9573	Net-Works Apple Juice, Drien, IL	
☐ 312-963-5384	Net-Works Apple Net, Chicago, IL	
☐ 312-935-3091	Net-Works Apple-Technical, Chicago, IL	
☐ 312-882-9237	Net-Works Chicago, IL	
☐ 312-323-3741	Net-Works Chipmunk, Hinsdale, IL	24h
☐ 312-255-6489	Net-Works CLAH, Chicago, IL	
☐ 312-627-5138	Net-Works Death Star, Oakbrook, IL	24h
☐ 312-998-5066	Net-Works Micro Ideas, Glenview, IL	
☐ 312-935-2933	Net-Works Pirate's Ship, IL	
☐ 312-393-4755	Net-Works RJNET, Warnville, IL	
☐ 312-441-6957	Outpost	
☐ 312-648-4867	Online Omega, Chicago, IL	24h
☐ 312-397-8308	OS-9 6809 BBS, Palatine	
☐ 312-359-9450	PBBS Co-operative Comp SVC, Palatine, IL	24h
☐ 312-397-0871	PET BBS Commodore, Chicago, IL	24h
☐ 312-373-8057	PMS Chicago, IL	24h
☐ 312-964-6513	PMS Downers Grove/Srt, Downers Grove, IL	
☐ 312-295-6926	PMS I.A.C., Lake Forest, IL	24h
☐ 312-876-0974	RBBS Milwaukee-Chicago Line	
☐ 312-647-7636	RCP/M A.B. Dick Co., Niles, IL	24h ★
☐ 312-326-4392	RCP/M Bridgeport, IL	24h
☐ 312-972-6979	RCP/M El Division, Argonne, IL	
☐ 312-469-2597	RCP/M Glen Ellyn, Chicago, IL	24h
☐ 312-967-0052	RCP/M Ham Radio, Morton Grove, IL	
☐ 312-252-2136	RCP/M Logan Square, Chicago, IL	24h
☐ 312-949-6189	RCP/M Nei, Chicago, IL	● ★
☐ 312-937-5639	RCP/M North Chicago, Chicago, IL	
☐ 312-251-0168	RCP/M North Side BBS, Chicago, IL	
☐ 312-789-0499	RCP/M RBBS Aims, Hinsdale, IL	24h
☐ 312-677-7140	South Pole	
☐ 312-623-2226	Waukegan Library, Waukegan, IL	

313

☐ 313-477-4471	ABBS, Detroit, MI	
☐ 313-978-8087	AMIS A.R.C.A.D.E., Sterling Heights, MI	24h
☐ 313-868-2064	AMIS M.A.C.E. Detroit, MI	24h
☐ 313-295-0783	Apple-Gram	24h
☐ 313-683-5076	Bullet-80, Waterford, MI	24h
☐ 313-465-9531	Comnet-80, Mt. Clemens, MI	★
☐ 313-856-3804	Crystal Castle	
☐ 313-764-1837	Davy Jones Locker	
☐ 313-644-3841	DWBBS	Ⓟ = BBS, UN = DW.BBS
☐ 313-474-5795	Electronic Odyssey, Detroit, MI	
☐ 313-453-9183	Monitor, Detroit, MI	
☐ 313-455-4227	Net-Works GBBS, Metro Detroit, MI	⚤
☐ 313-968-2645	Pirates Prison II	
☐ 313-846-6127	RCP/M CBBS Technical, Detroit, MI	24h ★
☐ 313-584-1044	RCP/M Detroit, MI	
☐ 313-759-6569	RCP/M MCBBS Keith Petersen, Royal Oak, MI	
☐ 313-559-5326	RCP/M RBBS Southfield, MI	24h
☐ 313-729-1905	RCP/M RBBS Westland, MI	
☐ 313-855-6006	Timewarp	
☐ 313-453-5146	T-Net Central Processing Unit, Detroit, MI	24h
☐ 313-855-6321	T-Net Special Corp	24h
☐ 313-775-1649	T-Net Twilight Phone, Warren, MI	24h
☐ 313-547-7903	Treasure Island	
☐ 313-533-0254	Westside Download, Detroit, MI	

314

☐ 314-535-3799	A.U.R.A. Atari 800, St. Louis, MO	24h
☐ 314-434-6187	Chambers of Xenobia	
☐ 314-625-4576	Commodore Communication, St. Louis, MO	24h
☐ 314-638-0644	Communitree, Golden Hind, St. Louis, MO	24h
☐ 314-645-1047	EMC-80, St. Louis, MO	
☐ 314-991-2744	Fantasy Island	
☐ 314-227-4312	Midwest, St. Louis, MO	⚤
☐ 314-432-7120	Net-Works Computer Station, MO	
☐ 314-968-7225	Net-Works Infoline, MO	
☐ 314-532-4652	Net-Works Forth Dimension, St. Louis, MO	

☐ 314-821-5826	Net-Works Space Age, MO	
☐ 314-994-9257	Net-Works St. Louis Exchange, MO	
☐ 314-576-4109	Pirates Emporium	

315

☐ 315-337-7720	Greene Machine, Rome, NY	
☐ 315-768-8153	Net-Works Elppa System, NY	

316

☐ 316-682-2113	Forum-80, Wichita, KS	24h ★

317

☐ 317-494-6643	FBBS #1, Purdue, IN	24h ★
☐ 317-326-3833	Net-Works, Greenfield, IN	24h
☐ 317-743-8667	Net-Works Von's Electronics, IL	
☐ 317-787-9881	Online, Indianapolis, IN	24h Ⓟ = pass id# = gues
☐ 317-255-5435	PET BBS AVC Comline, Indianapolis, IN	24h
☐ 317-787-5486	PMS, Indianapolis, IN	24h
☐ 317-742-7725	Viking Communications	

318

☐ 318-989-8537	Magic Kingdom	
☐ 318-988-1302	Net-Works Acadiana, LA	
☐ 318-861-1012	Net-Works Apple Gumbo, Shreveport, LA	24h
☐ 318-688-7078	NWLAIBMPCUG, Shreveport, LA	
☐ 318-237-3350	Star Link	
☐ 318-635-8660	TBBS, Shreveport, LA	24h
☐ 318-367-8860	USS Enterprise	

319

☐ 319-364-0811	CBBS Cedar Rapids, IA	24h
☐ 319-363-3314	RCP/M RBBS Hawkeye-PC, Cedar Rapids, IA	

401

☐ 401-521-2626	BBS Colornet, Providence, RI	● ★
☐ 401-738-5152	BBS Heathkit Store, Warwick, RI	●
☐ 401-272-1138	BBS Syslink, Providence, RI	24h
☐ 401-331-8450	Net-Works Computer City, RI	
☐ 401-751-5025	RCP/M Providence, Providence, RI	
☐ 401-944-4689	RI Tandy Users Group, Cranston, RI	24h
☐ 401-521-1998	R.I.A.M.I.S. Atari, Providence, RI	24h
☐ 401-456-8250	R.I.C.A.M.I.S., Kingston, RI	24h

402

☐ 402-476-1177	ABBS Linx, Lincoln, NE	24h dl
☐ 402-339-7809	ABBS, Omaha, NE	
☐ 402-571-8942	Dial-Your-Match #23, Omaha, NE	℘
☐ 402-734-4748	Mages Inn, Omaha, NE	24h
☐ 402-292-9598	OACPM Omaha, NE	24h
☐ 402-292-6184	Trade-80, Omaha, NE	

403

☐ 403-320-6923	Lethbridge Gaming System, Lethbridge, AB	
☐ 403-454-6093	RCP/M Dave Mccrady, Edmonton, AB, CAN	24h ★
☐ 403-482-6854	RCP/M RBBS Computron, Edmonton, AB, CAN	24h

404

☐ 404-256-1549	ABBS #X, Atlanta, GA	
☐ 404-790-8614	ABBS Baileys Computer Store, Augusta, GA	
☐ 404-252-4146	BBS IBM Hostcomm, Atlanta, GA	
☐ 404-294-6879	BBS IBM PC, Atlanta, GA	
☐ 404-252-9438	BBS IBM PC, Atlanta, GA	24h
☐ 404-461-9686	Bullet-80, Fayetteville, GA	
☐ 404-394-4220	CBBS, Atlanta, GA	24h
☐ 404-982-9627	Conference-Tree, Atlanta, GA	24h
☐ 404-279-5392	Forum-80, Augusta, GA	
☐ 404-733-3461	Net-Works Ags, Augusta, GA	24h
☐ 404-926-4318	Remote Northstar, Atlanta, GA	24h
☐ 404-962-0616	Telemessage-80, Atlanta, GA	

406

☐ 406-443-2768	RCP/M RBBS Helena Valley, Helena, MT	

408

☐ 408-259-7194	Appler HQ	
☐ 408-253-5216	AMIS Grafex, Cupertino, CA	
☐ 408-298-6930	AMIS IBBBS, San Jose, CA	
☐ 408-942-6975	AMIS TABBS, Sunnyvale, CA	
☐ 408-267-7399	Bird House, San Jose, CA	
☐ 408-980-0276	Buccaneer's Harbor	
☐ 408-475-7101	Conference-Tree, Berkeley, CA	
☐ 408-688-9629	Mines of Moria II, Aptos, CA	
☐ 408-227-5416	Net-Works Computer Emporium, CA	
☐ 408-996-7464	Net-Works The Dragon's Lair, CA	
☐ 408-688-9629	PMS Santa Cruz, Aptos, CA	24h
☐ 408-263-2588	RCP/M Colossal Oxgate, San Jose, CA	
☐ 408-378-8733	RCP/M Dbase II, San Jose, CA	24h
☐ 408-867-1243	RCP/M Oxgate 001, Saratoga, CA	24h ★
☐ 408-238-9621	RCP/M RBBS Datatech 007, San Jose, CA	24h
☐ 408-732-9190	RCP/M RBBS Datatech 010, Sunnyvale, CA	
☐ 408-287-5901	RCP/M RBBS San Jose Oxgate, San Jose, CA	24h
☐ 408-246-5014	RCP/M, Silicon Valley, CA	24h
☐ 408-730-8733	RCP/M, Sunnyvale, CA	●
☐ 408-739-5370	Shoalin Temple, Sunnyvale, CA	
☐ 408-867-4455	Split Infinity, Saratoga, CA	
☐ 408-338-9511	Stewart II	

409

☐ 409-846-2900	Net-Works Apple Seed, College Station, TX	24h
☐ 409-233-7943	PMS Gulfcoast, Freeport, TX	24h
☐ 409-845-0509	RCP/M Oxgate College Station, TX	24h
☐ 409-765-8866	The Treasure	

412

☐ 412-822-7176	CBBS PACC, Pittsburgh, PA	24h

414

☐ 414-637-9990	ABBS Colortron Computer, Racine, WI	24h
☐ 414-628-4352	Apparitions Cove	
☐ 414-353-1185	Atari Music Machine	
☐ 414-273-3434	Auto-Net, Milwaukee, WI	24h
☐ 414-483-4578	BBS SUE, Milwaukee, WI	
☐ 414-259-9475	BIG Top Games System, Milwaukee, WI	
☐ 414-241-8364	CBBS MAUDE, Milwaukee, WI	24h
☐ 414-679-9103	Commodore Up/Dnload Line	3pm-10pm
☐ 414-255-1222	Computer Palace, Milwaukee, WI	10am-10pm wknds
☐ 414-476-8722	Coco-Mug	24h
☐ 414-543-3333	Color-80, Milwaukee, WI	24h
☐ 414-672-6053	DataTech	24h
☐ 414-421-2863	Demon's Realm	6pm-6am
☐ 414-282-0501	Dragons Lair, Milwaukee, WI	
☐ 414-835-1754	E.S.C.A.P.E	(private)
☐ 414-964-5160	EXEC-PC	24h
☐ 414-282-4181	Generic, Milwaukee, WI	(private)
☐ 414-255-9645	H.A.U.S.E., Milwaukee, WI	7pm-7am
☐ 414-224-6930	Marquette	(private)
☐ 414-353-2402	Midnight Star	10pm-1pm
☐ 414-377-3878	Midwest Software Library	5pm-6am
☐ 414-327-5300	Milwaukee Express, Milwaukee, WI	24h $
☐ 414-281-0545	Milwaukee Tribune, Milwaukee, WI	24h
☐ 414-774-8478	Mini-Board	wknds
☐ 414-727-3637	Net-Works Lab-Works, WI	
☐ 414-554-9520	PET BBS S.E.W.P.U.G., Racine, WI	24h
☐ 414-784-0830	Radio Free Milwaukee, Milwaukee, WI	24h
☐ 414-462-2225	Rogue Moon	Fri & Sat 6pm-10am
☐ 414-476-8010	RSTS	(private)
☐ 414-762-6411	S.U.E	24h $
☐ 414-281-0545	TBBS Canopus, Milwaukee, WI	24h
☐ 414-649-8326	TEAM (TIBBS)	24h
☐ 414-542-2102	TeleCommunicator's Edge, Milwaukee, WI	
☐ 414-282-9308	The Connection, Milwaukee, WI	24h
☐ 414-541-0224	The Milwaukee BBS, Milwaukee, WI	24h
☐ 414-272-0369	Traders Alley, Milwaukee, WI	24h $
☐ 414-271-7580	Vanmil, Milwaukee, WI	24h
☐ 414-781-8653	Whizzz...'s Warez (AE)	

415

☐ 415-469-8111	ABBS South Of Market, San Francisco, CA	℘
☐ 415-895-8980	ATATCOM/80, San Leandro, CA	24h
☐ 415-658-2919	CBBS Lambda, Berkeley, CA	℘
☐ 415-357-1130	CBBS Proxima, Berkeley, CA	
☐ 415-820-0711	Chthon	
☐ 415-538-3580	Conference-Tree, Hayward, CA	
☐ 415-861-6489	Conference-Tree, San Francisco, CA	
☐ 415-626-9427	Conference-Tree, San Francisco, CA	
☐ 415-332-8115	Conference-Tree, Sausalito, CA	
☐ 415-651-4147	Connection-80, Fremont, CA	24h
☐ 415-522-1986	Dataworx	
☐ 415-991-4911	Dial-Your-Match #17	℘
☐ 415-467-2588	Dial-Your-Match #8, San Francisco, CA	℘
☐ 415-488-9145	Download-80 Mojo's, Forest Knolls, CA	24h ★
☐ 415-552-7671	Drummer	℘
☐ 415-348-2139	Forum-80, San Mateo, CA	★
☐ 415-897-2783	Greene Machine Golden State BBS, Novato, CA	
☐ 415-674-0660	Human & Wisdom	
☐ 415-481-0252	IBM PC No-name, San Lorenzo, CA	24h ★
☐ 415-522-6441	Litterbox	
☐ 415-565-3037	Living BBS, Education SIG	
☐ 415-352-8442	Motherboard, San Leandro, CA	
☐ 415-585-6334	Net-Works Apple Corps, San Francisco, CA	
☐ 415-482-2823	Night Owl	
☐ 415-775-2384	Pirates Bay	
☐ 415-924-6282	Pirates Warehouse	
☐ 415-462-7419	PMS Pleasanton, CA	24h
☐ 415-851-3453	PMS Portola Valley, CA	24h
☐ 415-490-7878	PMS Redington Group, Fremont, CA	24h
☐ 415-595-0541	RCP/M RBBS Datatech 001, San Carlos, CA	24h ★
☐ 415-461-7726	RCP/M RBBS, Larkspur, CA	24h
☐ 415-383-0473	RCP/M RBBS, Marin County, CA	24h
☐ 415-965-4097	RCP/M RBBS Piconet, Mountain View, CA	
☐ 415-552-9968	RCP/M Rich & Famous, San Francisco, CA	24h
☐ 415-941-1990	Realm of the Rogues	
☐ 415-452-0350	Sunrise Omega-80, Oakland, CA	
☐ 415-895-0699	System/80, San Leandro, CA	
☐ 415-490-8083	TBBS Noah's Ark, Fremont, CA	24h ℘
☐ 415-845-4812	Winner's Circle	

416

☐ 416-622-2462	Atari Info-System, Toronto, ON, CAN	24h
☐ 416-499-7023	BBS IBM Hostcomm, Toronto, ON, CAN	24h Ⓟ
☐ 416-487-5833	Bradley Brothers BBS, Toronto, ON, CAN	24h $
☐ 416-481-9047	Bradley Brothers BBS Download, Toronto, ON, CAN	24h $
☐ 416-265-3227	Bull 80, Toronto, ON, CAN	7:30pm-8am, 24h wknds
☐ 416-423-3265	Bull BBS (ETI Magazine), Toronto, ON, CAN	℘
☐ 416-461-2110	CBBS, Toronto, ON, CAN	24h
☐ 416-366-2069	CFTR BBS, Toronto, ON, CAN	6pm-9am
☐ 416-743-6221	Coco-Nut, Toronto, ON, CAN	24h
☐ 416-767-0412	Colour 80, Toronto, ON, CAN	6pm-9am
☐ 416-723-6500	Commodore 64 BBS, Oshawa, ON, CAN	
☐ 416-683-2226	Computer Camp BBS	5pm-9am
☐ 416-633-0185	Comspec BBS, Downsview, ON, CAN	
☐ 416-421-8930	Dr. Phobos Dating BBS, Toronto, ON, CAN	24h
☐ 416-921-4013	Exceltronics, Toronto, ON, CAN	24h
☐ 416-439-0065	Games BBS, Scarborough, ON, CAN	7pm-9am
☐ 416-482-2823	G.E. Nightowl, Toronto, ON, CAN	24h
☐ 416-877-0933	Georgetown HAM Radio BBS, Georgetown, ON, CAN	
☐ 416-278-3267	Infoport, Port Credit, ON, CAN	24h
☐ 416-762-1820	Insane Asylum, Toronto, ON, CAN	10pm-8am
☐ 416-978-6893	Medical Net-Works, Toronto, ON, CAN	7pm-9am
☐ 416-782-9686	Micro 80, Toronto, ON, CAN	8pm-8am
☐ 416-728-6574	Motor City BBS, Oshawa, ON, CAN	
☐ 416-445-6696	Net-Works, Toronto, ON, CAN	24h
☐ 416-683-3733	Net-Works, Toronto, ON, CAN	24h $
☐ 714-633-5240	Nortec BBS, Toronto, ON, CAN	24h
☐ 416-484-9663	OSBOARD, Toronto, ON, CAN	24h

☐ 416-624-5431 PET BBS PSI Wordpro, Mississauga, ON, CAN 24h
☐ 416-782-9534 PET BBS TPUG, Toronto, ON, CAN 24h ®
☐ 416-445-5192 PMS Logic Inc., North York, ON, CAN 24h $
☐ 416-335-6620 RCP/M HAPN Hamilton, ON, CAN 24h
☐ 416-232-0442 RCP/M Mississauga HUG, Mississauga, ON, CAN 24h ★
☐ 416-232-0269 RCP/M System One, Mississauga, ON, CAN 24h ★ $
☐ 416-231-1262 RCP/M System Two, Mississauga, ON, CAN 24h ★ $
☐ 416-884-6198 RTC BBS, Richmond Hill, ON, CAN 8pm-9am
☐ 416-839-3260 Superboard, Pickering, ON, CAN 9pm-8am
☐ 416-232-2644 THUG, Mississauga, ON, CAN 7pm-7am
☐ 416-451-7137 TMUG, Brampton, ON, CAN
☐ 416-839-8274 TRS-80 BBS, Pickering, ON, CAN
☐ 416-668-1851 TRS-80 BBS, Whitby, ON, CAN
☐ 416-445-1725 Twilight Comm, North York, ON, CAN

419

☐ 419-531-3845 ABBS Computer Store, Toledo, OH
☐ 419-867-9777 Toledo Apple Users BBS, Toledo, OH 24h

501

☐ 501-372-0576 PBBS Arc-Net, Little Rock, AR 24h
☐ 501-646-0197 PMS Ft. Smith Comp. Club, Ft. Smith, AK
☐ 502-459-5531 Net-Works Assembly Line, Louisville, KY ●
☐ 502-423-0695 Net-Works Baud-Ville, Louisville, KY ●

503

☐ 503-646-5510 CBBS, Portland, OR 24h
☐ 503-535-6883 Forum-80, Medford, OR 24h
☐ 503-635-7205 Freebooter's Archives
☐ 503-655-6009 Net-Works Oregon City, OR
☐ 503-641-2798 OARCS, Portland, OR
☐ 503-689-2655 PMS Computer Solutions, Eugene, OR 24h
☐ 503-245-2536 PMS, Portland, OR 24h
☐ 503-641-7276 RCP/M, Beaverton, OR 24h
☐ 503-621-3193 RCP/M Chuck Forsberg, OR 24h ★
☐ 503-649-7814 West Side Network, Portland, OR

504

☐ 504-889-2241 American Networks #2, Metarie, LA 24h ★
☐ 504-273-3116 CBBS, Baton Rouge, LA 24h
☐ 504-831-3589 Micro Phone
☐ 504-454-6688 Net-Works Crescent City, LA
☐ 504-291-4970 Trading Post

506

☐ 506-357-5668 TRS-80 BBS, Oromocto, NB, CAN

512

☐ 512-442-1116 Austin Party Board, Austin, TX 24h
☐ 512-578-5833 Conference-Tree, Victoria, TX
☐ 512-623-6123 Net-Works Alamo City, TX
☐ 512-494-0285 SATUG BBS, San Antonio, TX
☐ 512-443-3084 The Diner, Austin, TX
☐ 512-477-2672 The Paradise
☐ 512-441-9429 Thieve's Den
☐ 512-385-1102 TBBS, Austin, TX 24h

513

☐ 513-871-8901 Cook's Galley
☐ 513-223-3672 Net-Works, Dayton, OH
☐ 513-671-2753 PMS, Cincinnati, OH
☐ 513-489-0149 RCP/M RBBS, Cincinnati, OH ●
☐ 513-435-5201 RCP/M W. Carrolton, Dayton, OH 24h
☐ 513-863-7681 XBBS, Hamilton, OH 24h

514

☐ 514-622-1274 Connection-80, Laval Bele, Laval, PQ, CAN 24h
☐ 514-327-5764 Distra-Soft, Montreal, PQ, CAN 24h
☐ 514-931-0458 Online Computerland, Montreal, PQ, CAN 24h
☐ 514-332-3443 Pirates Brigade, Montreal, PQ, CAN

515

☐ 515-279-8863 Net-Works Computer Emporium, IA

516

☐ 516-698-4008 ABBS Pirates Cove, Long Island, NY
☐ 516-621-9296 Adventure BBS
☐ 516-561-6590 CBBS Lica Limbs, Long Island, NY 24h
☐ 516-334-3134 CBBS Long Island, NY 24h
☐ 516-775-5700 Compost
☐ 516-588-5836 Connection-80, Centereach, NY
☐ 516-482-8491 Connection-80, Great Neck, NY 24h
☐ 516-328-8204 Hardware Haven
☐ 516-367-8172 Haunted Mansion
☐ 516-627-9048 Net-Works Pirate's Trek
☐ 516-935-2481 Plover Net
☐ 516-751-5639 RCP/M Mid-Suffolk, Long Island, NY ●
☐ 516-293-8659 Ware-House II

517

☐ 517-339-3367 Connection-80, Lansing, MI

518

☐ 518-346-3596 Capital City BBS, Albany, NY 24h
☐ 518-235-9073 Cohoes Forum, Cohoes, NY
☐ 518-370-8343 Nibble One, Schenectady, NY

601

☐ 601-264-2361 Bullet-80, Hattiesburg, MS 24h
☐ 601-992-1918 Remote Apple, Jackson, MS 24h

602

☐ 602-898-0891 ABBS Phoenix, AZ
☐ 602-996-9709 A.C.C.E.S.S. Phoenix, AZ 24h
☐ 602-957-4428 A.C.C.E.S.S. Phoenix, AZ 24h ★
☐ 602-275-6644 A.C.C.E.S.S. Phoenix, AZ
☐ 602-274-5964 A.C.C.E.S.S. Phoenix, AZ
☐ 602-998-9411 A.C.C.E.S.S. Scotsdale, AZ 24h
☐ 602-246-1432 BBS Apollo, Phoenix, AZ 24h
☐ 602-952-1382 Blax-80 BBS, Phoenix, AZ 24h
☐ 602-275-6644 Call-A-Lawyer, Phoenix, AZ 24h
☐ 602-746-3956 CBBS TSG, Tucson, AZ 24h
☐ 602-931-1829 Conference-Tree, Phoenix, AZ 24h
☐ 602-956-5021 Creepy Corridors, Phoenix, AZ ●
☐ 602-890-0972 Diamond III, Phoenix, AZ 24h
☐ 602-458-3850 Forum-80, Sierra Vista, AZ 24h
☐ 602-967-4529 Genesys, Phoenix, AZ 24h
☐ 602-726-7533 Greene Machine, Yuma, AZ 24h ★
☐ 602-251-8538 Magic Lantern
☐ 602-938-4508 Microsystems, Phoenix, AZ 24h
☐ 602-952-2018 Omega, Phoenix, AZ 24h
☐ 602-833-0740 Stellar III, Phoenix, AZ 24h
☐ 602-861-4090 System-X, Phoenix, AZ –
☐ 602-991-0144 Garden Of Eden, Phoenix, AZ 24h
☐ 602-247-6034 Voyager, Phoenix, AZ

603

☐ 603-924-7920 Connection-80, Peterborough, NH
☐ 603-882-5041 Forum-80, Nashua, NH
☐ 603-436-3461 Net-Works, Portsmouth, NH
☐ 603-625-1919 Software Referral Service

604

☐ 604-437-7001 ABBS Vancouver, BC, CAN
☐ 604-682-6551 ABC Vancouver, BC, CAN
☐ 604-922-1336 Apple Perch
☐ 604-271-3354 Basically BBS, Vancouver, BC, CAN
☐ 604-562-9515 CBBS, Prince George, BC, CAN
☐ 604-687-2640 CBBS Vancouver, BC, CAN 24h
☐ 604-430-8233 Heath BBS, Vancouver, BC, CAN
☐ 604-591-6975 Message 80, Surrey, BC, CAN 24h
☐ 604-224-2337 Microstat, BC, CAN
☐ 604-584-1047 Pacific Blue, BC, CAN
☐ 604-937-0906 RCP/M CBBS Frog Hollow, Vancouver, BC, CAN 24h
☐ 604-584-2543 RCP/M RBBS, Surrey, BC, CAN 24h
☐ 604-873-4007 RCP/M Vancouver, BC, CAN 24h

607

☐ 607-797-6416 RCP/M SJBBS, Johnson City, NY ●
☐ 604-438-2468 Satyricomp, BC, CAN
☐ 604-584-2731 SMUG, BC, CAN

608

☐ 608-251-8538 AMIS Magic Lantern, Madison, WI
☐ 608-262-4939 BBS IBM PC, Madison, WI 24h

609

☐ 609-228-1149 ABBS, Turnersville, NJ
☐ 609-468-5293 RATS, Wenonah, NJ
☐ 609-468-3844 RATS, Wenonah, NJ #2
☐ 609-896-2436 T-Net Delta Connection 24h

612

☐ 612-472-3985 ABBS Calvary Mission Church, Minneapolis, MN 24h †
☐ 612-724-7066 BBS The Safehouse, Minneapolis, MN 24h
☐ 612-377-7747 Captain's Log
☐ 612-423-5016 CBBS, Rosemont, MN
☐ 612-854-9691 Conference-Tree, Minneapolis, MN
☐ 612-938-7535 Deep Thot
☐ 612-753-3082 MCMS Goliath, Minneapolis, MN
☐ 612-533-1957 MCMS NC Software, Minneapolis, MN 24h
☐ 612-546-1013 On-Target
☐ 612-825-5852 Pirates Island
☐ 612-929-6699 PMS, Minneapolis, MN 24h
☐ 612-929-8966 PMS, Twin Cities, Minneapolis, MN
☐ 612-454-6209 The Grapevine

613

☐ 613-725-2243 ABBS Compumart, Ottawa, ON, CAN
☐ 613-820-4646 Forum-80, Ottawa, ON, CAN
☐ 613-236-3009 CBBS Ottawa, ON, CAN
☐ 613-236-3009 ETW BBS, Ottawa, ON, CAN

614

☐ 614-475-9791 Applecrackers, Columbus, OH 24h
☐ 614-532-6920 Bullet-80, Ironton, OH
☐ 614-423-4422 Ohio Valley BBS
☐ 614-272-2227 RCP/M CBBS, Columbus, OH 24h
☐ 614-837-3269 RCP/M RBBS, Pickerington, OH

615

☐ 615-297-6037 Knight Line

616

☐ 616-382-0101 ABBS Computer Room, Kalamazoo, MI
☐ 616-241-1971 AMIS G.R.A.S.S., Grand Rapids, MI 24h
☐ 616-457-1840 Connection-80 W. Micn. Micro Group, MI 24h
☐ 616-531-0890 HBBS Heath/Zenith, Grand Rapids, MI ★
☐ 616-693-2648 RS-CPM, Clarksville, MI

617

☐ 617-876-4885 AMIS Starbase 12, Philadelphia, PA
☐ 617-353-9312 BBS IBM PC Computer Society, Boston, MA ●
☐ 617-423-6985 BOSTON Information Exchange, Boston, MA 24h ★
☐ 617-266-7789 Bullet-80, Boston, MA 24h ★
☐ 617-279-0522 Captain Flint's Quarterdeck
☐ 617-646-3610 CBBS, Boston, MA 24h
☐ 617-683-2119 CBBS Lawrence General Hospital, Boston, MA
☐ 617-752-7284 CBBS Microstar, Worcester, MA
☐ 617-865-1264 Davy Jones Locker, Lexington, MA
☐ 617-334-6369 Dial-Your-Match #18 ⚤
☐ 617-692-3973 Forum-80, Westford, MA
☐ 617-332-5017 Hanger 19
☐ 617-256-1446 Net-Works Microbbs, Chelmsford, MA
☐ 617-494-1985 Net-Works Pirate's Harbor, MA
☐ 617-720-3600 Net-Works Pirate's Harbor, Boston, MA
☐ 617-891-1349 Pirates Chest
☐ 617-863-1237 Pirates Hideout, Lexington, MA
☐ 617-965-2436 Post Office

☐ 617-767-1303	PMS Apple Guild, Weymouth, MA	24h
☐ 617-774-7516	PMS Computer City, Danvers, MA	
☐ 617-862-0781	RCP/M Superbrain, Lexington, MA	24h ★
☐ 617-863-0282	TermExec Newsletter, Lexington, MA	
☐ 617-443-7428	Trading Post II	
☐ 617-235-5082	Visiboard, Wellesley, MA	
☐ 617-326-4812	Westwood BBS	

618

☐ 618-692-0742	Net-Works Asylum, IL	
☐ 618-877-2904	Net-Works, Granite City, IL	
☐ 618-254-6074	Net-Works Harpos Bar & Grill, IL	
☐ 618-466-9497	Net-Works NAGS, IL	
☐ 618-345-6638	Net-Works Warlock's Castle, St. Louis, MO	
☐ 618-451-1041	Sattelite/Cable Net	
☐ 618-797-0656	Skull Island V	
☐ 618-234-4243	TPS Network	

619

☐ 619-691-8367	CVBBS, San Diego, CA	24h
☐ 619-434-4600	Dial-Your-Match #11, Carlsbad, CA	24h ☎
☐ 619-748-8746	Dial-Your-Match #33, Poway, CA	24h ☎
☐ 619-692-1961	Online Saba, San Diego, CA	24h
☐ 619-561-7271	P.DBMS, Lakeside, CA	24h ★
☐ 619-582-9557	PMS Computer Merchant, San Diego, CA	24h
☐ 619-271-8613	PMS Datel Systems Inc., San Diego, CA	24h
☐ 619-265-3428	PMS Ed Tech, San Diego, CA	
☐ 619-746-0667	PMS, Escondido, CA	●
☐ 619-579-7036	PMS Floppy House, San Diego, CA	24h
☐ 619-251-8538	PMS Floppy House	
☐ 619-578-2646	PMS Kid's Message System, San Diego, CA	24h
☐ 619-727-7500	PMS, San Marcos, CA	24h
☐ 619-561-7277	PMS, Santee, CA	24h ml
☐ 619-256-3914	RCP/M, Barstow, CA	24h ★
☐ 619-273-4354	RCP/M RBBS, San Diego, CA	24h ★
☐ 619-461-0111	RCP/M RBBS SDCS Hec#04, La Mesa, CA	●
☐ 619-236-0742	RCP/M RBBS SDCS, San Diego, CA	24h
☐ 619-534-1547	RCP/M, San Diego, CA	24h ★

701

☐ 701-746-4959	Net-Works Armadillo, Grand Forks, ND	

702

☐ 702-870-9986	Comnet-80, Las Vegas, NV	★
☐ 702-362-3609	Forum-80, Las Vegas, NV	24h
☐ 702-878-9106	PMS Century 23, Las Vegas, NV	24h
☐ 702-826-7277	Signon, Reno, NV	★ Ⓟ = free

703

☐ 703-471-0610	ABBS Software Sorcery, Herndon, VA	24h ★
☐ 703-978-9754	BBS, Annandale, VA	
☐ 703-978-9592	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 703-978-0921	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 703-591-5120	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 703-425-9452	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 703-425-7229	BBS IBM Hostcomm, Springfield, VA	24h
☐ 703-560-0979	BBS IBM PC, Annandale, VA	24h
☐ 703-680-5220	BBS IBM PC, Dale City, VA	24h
☐ 703-759-5049	BBS IBM PC, Great Falls, VA	24h ★
☐ 703-560-7803	BBS IBM PC, Vienna, VA	24h
☐ 703-823-5210	Carrier 2, Alexandria, VA	
☐ 703-734-1387	CBBS Amrad, Washington, DC	24h
☐ 703-360-3812	C-HUG Bulletin Board, Fairfax, VA	24h
☐ 703-670-5881	Forum-80, Prince William County, VA	24h
☐ 703-360-5439	Future Tech, Alexandria, VA	24h
☐ 703-471-0611	Magus, Herndon Va	24h
☐ 703-644-1665	Pirates Trove	
☐ 703-323-4791	Pirates Trove III	
☐ 703-379-0303	Potomac Micro Magic Inc., Falls Church, VA	24h
☐ 703-536-3769	RCP/M, Arlington, VA	●
☐ 703-524-2549	RCP/M CBBS RLP, Maclean, VA	24h
☐ 703-342-1800	Star City	
☐ 703-765-2161	Switchboard, Alexandria, VA	24h
☐ 703-836-0384	TCUG BBS, Washington, DC	24h
☐ 703-328-4443	WCCC	

704

☐ 704-364-5245	ABBS, Charlotte, NC	24h
☐ 704-365-4311	BBS IBM PC, Charlotte, NC	24h
☐ 704-373-7966	WAPABBS, Charlotte, NC	24h

707

☐ 707-585-3586	BBS Express	
☐ 707-539-6471	Byte The Bulletin	
☐ 707-527-5908	Dual BBS-16, Santa Rosa, CA	
☐ 707-528-3462	Net-Works Micro-Sys, CA	
☐ 707-538-9124	SRTRS-80 Grape Vine BBS, Napa Valley, CA	24h
☐ 707-422-7256	RCP/M RBBS, Fairfield, CA	
☐ 707-257-6502	RCP/M RBBS, Napa Valley, CA	24h
☐ 707-576-1478	Software 1st BBS	
☐ 707-523-1736	SRCC ABBS, Santa Rosa, CA	
☐ 707-996-2427	Tel-Com	

712

☐ 712-368-2651	Bullet-80, Holstein, IA	

713

☐ 713-468-3122	Apple Crunch, Houston, TX	
☐ 713-890-0310	BBS IBM Hostcomm, Houston, TX	24h
☐ 713-661-5428	BBS MCUA, Houston, TX	24h
☐ 713-444-7041	Compuque-80, Houston, TX	24h ★
☐ 713-376-6382	Cyrus Dimension	
☐ 713-556-1531	Dial-Your-Match #12, Houston, TX	24h ☎
☐ 713-783-4136	Dial-Your-Match #24, Houston, TX	☎
☐ 713-471-4131	Doc Board, Houston, TX	
☐ 713-530-5249	Fantasy Voyage	
☐ 713-444-7098	GABBS, Armadillo Media, Houston, TX	24h
☐ 713-455-6502	GABBS, Houston, TX	24h
☐ 713-932-1124	Jolly Roger #2, Houston, TX	
☐ 713-782-5706	Net-Works Briar-Net, Houston, TX	24h
☐ 713-468-0174	Net-Works Jolly Roger, Houston, TX	24h
☐ 713-864-4672	Net-Works Micro Design, Houston, TX	●
☐ 713-871-8577	Net-Works Mines Of Moria, Houston, TX	24h
☐ 713-974-5258	Net-Works Pirate's Palace, Houston, TX	24h
☐ 713-333-2309	Net-Works The Dark Realm, Houston, TX	24h
☐ 713-354-4690	Net-Works The Inner Realm, Houston, TX	24h
☐ 713-777-8608	Net-Works The Shadow World, Houston, TX	24h
☐ 713-785-7996	Net-Works The System, Houston, TX	●
☐ 713-492-8700	Net-Works The Weekender, Houston, TX	24h
☐ 713-933-7353	Net-Works Zachary-Net, Houston, TX	24h
☐ 713-441-4032	PMBBS	
☐ 713-438-2247	RCP/M Blue Ridge, Missouri City, TX	24h
☐ 713-862-1624	RCP/M RBBS Pegasus, Houston, TX	24h
☐ 713-469-8893	RCP/M Satsuma, Houston, TX	● ★
☐ 713-522-3805	RCP/M Technical, Houston, TX	
☐ 713-497-5433	RIBBS, Houston, TX	
☐ 713-453-7931	SOBBS Poor Man's BBS, Houston, TX	24h
☐ 713-522-5516	SOBBS Test Mode, Houston, TX	
☐ 713-468-0198	Software House, Houston, TX	
☐ 713-568-6595	Space Voyage, Houston, TX	
☐ 713-442-7644	TBBS Exidy 2000, Houston, TX	24h ★
☐ 713-331-2599	TBBS Freelancin' Alvin, Houston, TX	24h ★
☐ 713-488-2003	TBBS Freelancin', Houston, TX	24h ★
☐ 713-944-6597	VIC-20 Online, Houston, TX	24h
☐ 713-495-1422	XIO, Houston, TX	●

714

☐ 714-583-3103	Adventurer's Tavern	
☐ 714-952-2110	Bullet-80 Orange County, Anaheim, CA	
☐ 714-644-7942	Bullet-80 Pirate Place, CA	
☐ 714-770-5052	Comnet-80, Laguna Hills, CA	
☐ 714-359-3189	Comnet-80, Riverside, CA	★
☐ 714-877-2253	Comnet-80, Riverside, CA	★
☐ 714-983-9923	Computers For Christ, Ontario, CA	24h
☐ 714-974-9788	Dimension-80, Orange, CA	
☐ 714-841-5321	Dune	
☐ 714-532-4521	Flipper's, Garden Grove, CA	
☐ 714-354-8004	Greene Machine Riverside, CA	
☐ 714-545-7359	IDBN Info-Net, Costa Mesa, CA	
☐ 714-551-4336	Irvine Line, Irvine, CA	
☐ 714-823-1451	Net-Works Apple Jacks, CA	
☐ 714-633-5240	North Orange County Computer Club, Orange, CA	
☐ 714-530-8226	OCTUG Orange County, Garden Grove, CA	
☐ 714-537-7913	Orange County Data Exchange, Garden Grove, CA	
☐ 714-545-8100	Pig Sty, Costa Mesa, CA	
☐ 714-772-8868	PMS **if**, Anaheim, CA	24h
☐ 714-524-1228	RACS V, Fullerton, CA	
☐ 714-774-7860	RCP/M CBBS Anahug, Anaheim, CA	24h
☐ 714-534-1547	RCP/M RBBS GFRN Data Exchange, Garden Grove, CA	24h ★
☐ 714-535-7527	The Simarillion, Garden Grove, CA	
☐ 714-547-6220	Verga 80, Costa Mesa, CA	

716

☐ 716-244-9531	CBBS Rams, Rochester, NY	
☐ 716-425-1785	RCP/M RBBS, Rochester, NY	24h ★

717

☐ 717-586-2112	Bullet-80, Clarks Summit, PA	

802

☐ 802-879-4981	ABBS Vermont, Essex Junction, VT	24h
☐ 802-862-7023	ST80-CC Lance Micklus Inc., Burlington, VT	24h

803

☐ 803-771-0922	Compusystems, Columbia, SC	
☐ 803-552-1612	Forum-80, Charleston, SC	24h
☐ 803-548-0900	RCP/M RBBS Fort Mill, SC	24h

804

☐ 804-491-1437	Atari BBS, Virginia Beach, VA	24h
☐ 804-444-3392	NBBS Norfolk, VA	
☐ 804-898-7493	RCP/M Oxgate 007, Grafton, VA	24h
☐ 804-340-5246	Remote Northstar, Virginia Beach, VA	
☐ 804-285-0041	Skeleton Island	

805

☐ 805-522-4211	Apple-Net II, Santa Susana Knolls, CA	24h
☐ 805-496-0850	Computer Connection	
☐ 805-522-1789	Net-Works Visual Comm, CA	
☐ 805-492-3150	Pirates Phunhouse, Thousand Oaks, CA	
☐ 805-527-9321	RCP/M CBBS CP/M Net Simi Valley, CA	
☐ 805-527-2219	RCP/M Simi Valley, CA	●
☐ 805-492-5472	RCP/M Technical, Thousand Oaks, CA	24h ★
☐ 805-964-4115	Remote Northstar Santa Barbara, CA	
☐ 805-493-1152	Treasure Vault, Thousand Oaks, CA	

808

☐ 808-944-0562	CBBS Strictly Software, Honolulu, HI	
☐ 808-487-2001	Conference-Tree Computerland, Honolulu, HI	24h
☐ 808-524-6668	Net-Works Computer Market, Honolulu, HI	●
☐ 808-488-7756	Net-Works Computer Store, Honolulu, HI	
☐ 808-423-1593	Net-Works Hawaii Connection, Honolulu, HI	24h
☐ 808-521-7312	Net-Works Hawaii, Honolulu, HI	

809

☐ 809-781-0350	BBS Commodore, San Juan, PR	●

812

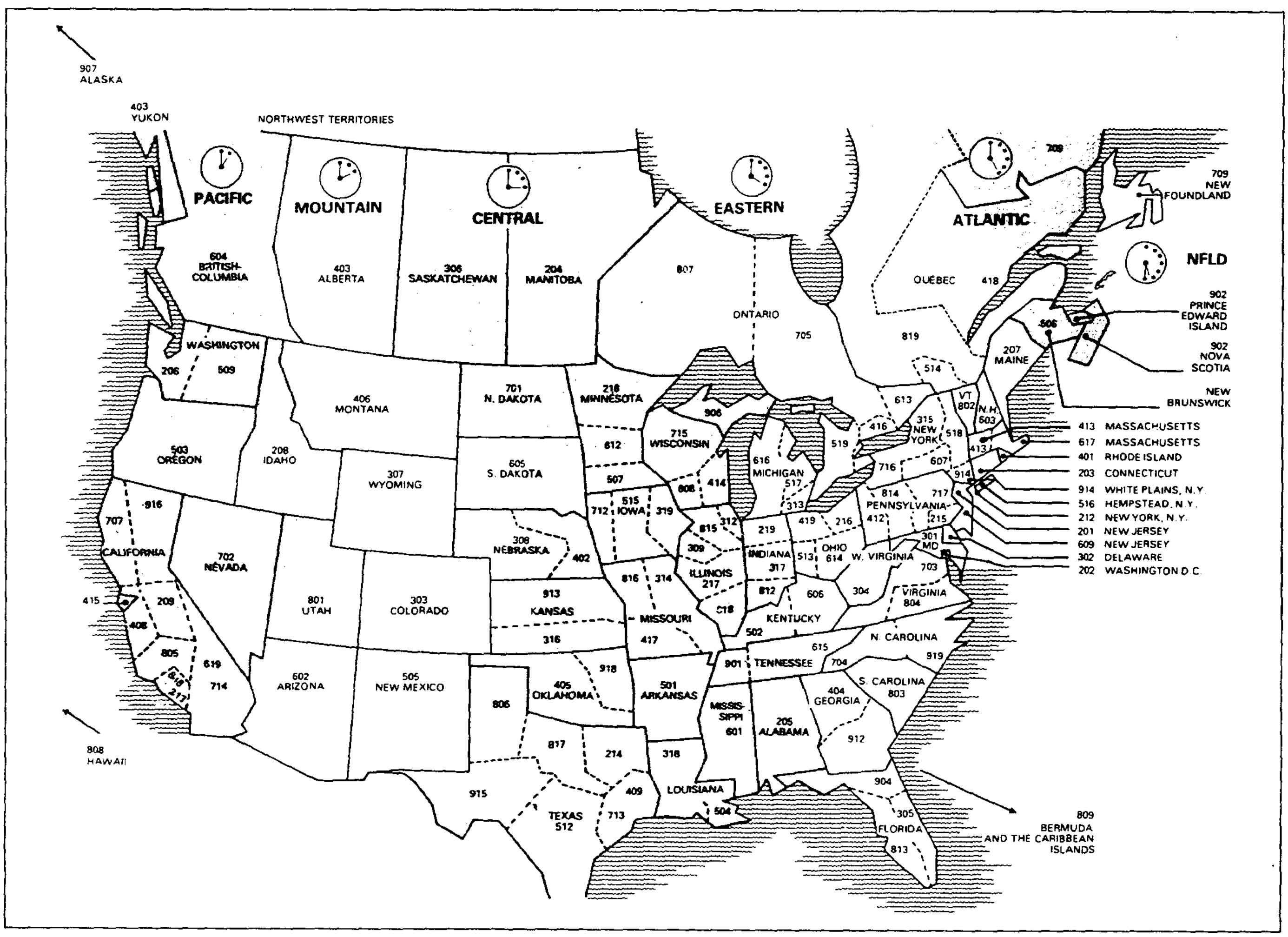
☐ 812-334-2522	CBBS Bloomington, IN	
☐ 812-858-5405	Net-Works Nick Naimo, Newburgh, IN	

813

☐ 813-884-1506	Access-80, Tampa, FL	24h
☐ 813-251-4095	Alpha, Tampa, FL	24h Ⓟ = tryit, ac# = abcd00
☐ 813-645-3669	Apollo's Chariot, Apollo, FL	
☐ 813-734-7103	Bradley Computer BBS	
☐ 813-885-6187	BSBB Tampa, FL	

☐	813-866-9945	CBBS St. Petersburg, FL	24h
☐	813-977-0989	Connection-80 Tampa, FL	
☐	813-875-3331	Micro Informer, Tampa, FL	
☐	813-391-5219	PET BBS Commodore, Largo, FL	
☐	813-831-7276	RCP/M RBBS Tampa, FL	
☐	813-381-2394	Remote Northstar Largo, FL	24h
☐	813-839-6746	Tecom-80, Tampa, FL	
		814	
☐	814-238-4857	RCP/M CUG-Node, PA State College	24h
☐	814-898-2952	Trade-80 Erie, PA	24h
		815	
☐	815-397-4176	Cider City	
☐	815-455-2406	Flynn's Games	
☐	815-838-1020	MCMS J.A.M.S. Lockport, IL	24h
		816	
☐	816-587-9543	BBS Atari Amis, Kansas City, MO	24h
☐	816-861-7040	Forum-80 Kansas City, MO	24h ★
☐	816-931-9316	Forum-80 Kansas City, MO	★
☐	816-483-2526	Net-Works ABC, Kansas City, MO	
☐	816-232-3153	Net-Works The Silver Tongue, ST. Joseph, MO	
☐	816-252-0232	PMS Apple Bits, Kansas City, MO	24h
		817	
☐	817-767-5847	Comnet-80 Wichita Falls, TX	
☐	817-665-3876	Dragonfire	
☐	817-261-4700	Net-Works Compushop FWA, TX	
☐	817-732-1787	Net-Works Computer Pro, Ft. Worth, TX	
☐	817-283-3886	Texas Connection	
		901	
☐	901-761-4743	ABBS Computer Lab, Memphis, TN	
☐	901-276-8196	Forum-80 Medical, Memphis, TN	24h
		904	
☐	904-243-1257	ABBS Fort Walton Beach, Destin, FL	
☐	904-477-8783	BBS Pensacola, FL	
☐	904-264-0335	Colour-80, Orange Park, FL	24h
☐	904-353-5227	Connection-80 Jacs, Jacksonville, FL	24h
☐	904-932-8271	Net-Works Beach BBS, Pensacola, FL	
☐	904-743-7050	PMS Seb Computer, Jacksonville, FL	
☐	904-725-4995	RCP/M RBBS Jug, Jacksonville, FL	24h ★
		907	
☐	907-225-6789	ABBS, Ketchikan, AK	
☐	907-344-5251	Conference-Tree, Anchorage, AK	
☐	907-278-4223	Net-Works Alaska	
☐	907-344-8558	PMS Anchorage, AK	

☐	907-337-1984	RCP/M Anchorage, AK	●
		912	
☐	912-233-0863	Dial-Your-Match #3	⚤
☐	912-439-7440	Trade-80, Albany, GA	24h
		913	
☐	913-676-3613	Experimental-80, Kansas City, MO	
☐	913-648-6071	Net-Works Leawood, KS	
☐	913-432-5544	Online Dickinsons Movie Guide, Mission, KS	24h
☐	913-677-1299	PMS Your Computer Connection, Kansas City, MO	●
☐	913-362-9583	RCP/M, Mission, KS	24h ★
☐	913-843-4259	RCP/M RBBS Alphanet, Lawrence, KS	●
☐	913-648-5301	Steve's BBS	24h
		914	
☐	914-634-1268	Net-Works Pirate's Lodge NY	
☐	914-592-5385	Nybbles-80, Elmsford, NY	
☐	914-725-4060	OSUNY, Scarsdale, NY	
☐	914-942-2638	RACS III	
☐	914-279-5693	RCP/M RBBS, Brewster, NY	●
☐	914-679-8734	RCP/M RBBS, Woodstock, NY	24h ★
☐	914-679-6559	RCP/M SJBBS, Bearsville, NY	24h
☐	914-359-1517	Sherwood Forest II	
☐	914-782-7605	ST80-PBB Monroe Camera Shop, Monroe, NY	
☐	914-623-4248	Teleport 64	
		915	
☐	915-565-9903	Bullet-80, El Paso, TX	24h
☐	915-755-1000	Forum-80, El Paso, TX	24h
☐	915-593-6655	Net-Works El Paso, TX	
☐	915-533-2202	RCP/M RBBS Comp. Tech. Assoc., El Paso, TX	24h
☐	915-598-1668	RCP/M RBBS, El Paso, TX	24h ★
		916	
☐	916-393-4459	Aviators Bulletin Board, Sacramento, CA	
☐	916-483-8718	RCP/M CBBS, Sacramento, CA	24h
		918	
☐	918-838-8698	Infoex-80, Tulsa, OK	24h
☐	918-749-0059	TBBS, Tulsa, OK	24h
		919	
☐	919-362-0676	Dial-Your-Match #20	⚤
		Foreign	
☐	613-762-5088	RCP/M CBBS Micom, Melbourne, VIC, Australia	24h
☐	1 0-997-1018	RCP/M Software Tools, Sydney, Australia	24h
☐	4-1 399-2136	CBBS, London, England	(European Standard)
☐	44 482859169	Forum-80, Hull, England	(Country Code = 011)



Bulletin Boards In Alphabetical Order

24h Denotes 24-hour operation ↔ **Multi-User System** **$ Pay System, Password Required** ⚤ **Sexually Oriented BBS**
● **Nighttime Operation** ★ **1200 Baud Allowed** Ⓟ **Password Required** † **Religious orientation**

A

☐ 404-256-1549	ABBS #X, Atlanta, GA	
☐ 216-745-7855	ABBS Akron Digital Group, Akron, OH	24h
☐ 206-935-9119	ABBS Apple Crate I, Seattle, WA	
☐ 206-244-5438	ABBS Apple Crate II, Seattle, WA	
☐ 201-864-5345	ABBS Apple-Mate, New York, NY	
☐ 404-790-8614	ABBS Baileys Computer Store, Augusta, GA	
☐ 305-486-2983	ABBS Byte Shop, Ft. Lauderdale, FL	
☐ 305-261-3639	ABBS Byte Shop, Miami, FL	
☐ 612-472-3985	ABBS Calvary Mission Church, Minneapolis, MN	24h †
☐ 201-835-7228	ABBS CCNJ, Pompton Plains, NJ	
☐ 704-364-5245	ABBS, Charlotte, NC	24h
☐ 312-882-2926	ABBS Code, Glen Ellyn, IL	24h
☐ 414-637-9990	ABBS Colortron Computer, Racine, WI	24h
☐ 613-725-2243	ABBS Compumart, Ottawa, ON, CAN	
☐ 213-829-1140	ABBS Computer Conspiracy, Santa Monica, CA	
☐ 301-730-0922	ABBS Computer Crossroads, Columbia, MD	
☐ 901-761-4743	ABBS Computer Lab, Memphis, TN	
☐ 616-382-0101	ABBS Computer Room, Kalamazoo, MI	
☐ 419-531-3845	ABBS Computer Store, Toledo, OH	
☐ 214-424-3862	ABBS Dallas Info Board, Dallas, TX	
☐ 303-759-2625	ABBS, Denver, CO	
☐ 313-477-4471	ABBS, Detroit, MI	
☐ 904-243-1257	ABBS Fort Walton Beach, Destin, FL	
☐ 312-475-4884	ABBS Gamemaster, Chicago, IL	24h
☐ 907-225-6789	ABBS, Ketchikan, AK	
☐ 402-476-1177	ABBS Linx, Lincoln, NE	24h
☐ 402-339-7809	ABBS, Omaha, NE	
☐ 213-459-6400	ABBS Pacific Palisades, Los Angeles, CA	
☐ 309-692-6502	ABBS, Peoria, IL	
☐ 602-898-0891	ABBS, Phoenix, AZ	
☐ 516-698-4008	ABBS Pirates Cove, Long Island, NY	
☐ 312-973-2227	ABBS Rogers Park, Chicago, IL	
☐ 703-471-0610	ABBS Software Sorcery, Herndon, VA	24h ★
☐ 415-469-8111	ABBS South Of Market, San Francisco, CA	⚤
☐ 214-960-7654	ABBS Teledunjon III, Dallas, TX	
☐ 214-631-7747	ABBS The Pulse, Dallas, TX	24h ⚤
☐ 609-228-1149	ABBS, Turnersville, NJ	
☐ 604-437-7001	ABBS, Vancouver, BC, CAN	
☐ 802-879-4981	ABBS Vermont, Essex Junction, VT	24h
☐ 312-475-5282	ABBS Video Adv. Movie Marquee, Evanston, IL	
☐ 305-848-3802	ABBS, West Palm Beach, FL	
☐ 604-682-6551	ABC Vancouver, BC, CAN	
☐ 301-267-7666	A.C.C.E.S.S., Annapolis, MD	24h
☐ 206-866-9043	A.C.C.E.S.S., Olympia, WA	24h
☐ 602-996-9709	A.C.C.E.S.S., Phoenix, AZ	24h
☐ 602-957-4428	A.C.C.E.S.S., Phoenix, AZ	24h ★
☐ 602-275-6644	A.C.C.E.S.S., Phoenix, AZ	
☐ 602-274-5964	A.C.C.E.S.S., Phoenix, AZ	
☐ 602-998-9411	A.C.C.E.S.S., Scotsdale, AZ	24h
☐ 201-891-7441	A.C.C.E.S.S., Wyckoff, NJ	24h
☐ 813-884-1506	Access-80, Tampa, FL	24h
☐ 213-537-3378	Access One, CA	
☐ 312-392-2403	ACS, Arlington Heights, IL	
☐ 312-445-1130	ACS, Chicago, IL	
☐ 516-621-9296	Adventure BBS	
☐ 714-538-3103	Adventurer's Tavern	
☐ 202-364-8617	Aladdin's Lamp	
☐ 301-881-0846	Alcatraz	
☐ 213-564-7636	All Night BBS, CA	
☐ 813-251-4095	Alpha, Tampa, FL	24h Ⓟ = tryit, ac# = abcd00
☐ 213-991-1604	Alpha Byte, CA	
☐ 303-333-1132	American BBS	
☐ 504-889-2241	American Networks #2, Metarie, LA	24h ★
☐ 313-978-8087	AMIS A.R.C.A.D.E., Sterling Heights, MI	24h
☐ 305-238-1231	AMIS Apogee, Miami, FL	
☐ 312-789-3610	AMIS, Clarendon Hills, IL	24h
☐ 616-241-1971	AMIS G.R.A.S.S., Grand Rapids, MI	24h
☐ 408-253-5216	AMIS Grafex, Cupertino, CA	
☐ 408-298-6930	AMIS IBBBS, San Jose, CA	
☐ 313-868-2064	AMIS M.A.C.E., Detroit, MI	24h
☐ 608-251-8538	AMIS Magic Lantern, Madison, WI	
☐ 617-876-4885	AMIS Starbase 12, Philadelphia, PA	
☐ 408-942-6975	AMIS TABBS, Sunnyvale, CA	
☐ 206-621-8665	Anchor CP/M	
☐ 201-790-5910	Aphrodite-E, Haledon, NJ	⚤
☐ 813-645-3669	Apollo's Chariot, Apollo, FL	
☐ 414-628-4352	Apparitions Cove	
☐ 206-525-5410	Apple Crate I, Seattle, WA	
☐ 713-468-3122	Apple Crunch, Houston, TX	
☐ 313-295-0783	Apple-Gram	24h
☐ 805-522-4211	Apple-Net II, Santa Susana Knolls, CA	24h
☐ 312-963-5384	Apple Juice	
☐ 604-922-1336	Apple Perch	
☐ 614-475-9791	Applecrackers, Columbus, OH	24h
☐ 408-259-7194	Appler HQ	
☐ 206-546-6239	ARBB, Seattle, WA	
☐ 301-587-2132	ARMUDIC Computer Age, Baltimore, MD	
☐ 202-276-8342	ARMUDIC, Washington, DC	
☐ 301-984-3772	ASCII	
☐ 312-674-2578	AT&T Phone Center	
☐ 804-491-1437	Atari BBS, Virginia Beach, VA	24h
☐ 416-622-2462	Atari Info-System, Toronto, ON, CAN	24h
☐ 415-895-8980	ATATCOM/80, San Leandro, CA	24h
☐ 414-353-1185	Atari Music Machine	
☐ 314-535-3799	A.U.R.A. Atari 800, St. Louis, MO	24h
☐ 303-343-8401	Aurora-Net	
☐ 512-442-1116	Austin Party Board, Austin, TX	24h
☐ 414-273-3434	Auto-Net, Milwaukee, WI	24h
☐ 916-393-4459	Aviators Bulletin Board, Sacramento, CA	
☐ 213-851-0780	Aware II, Los Angeles, CA	

B

☐ 604-271-3354	Basically BBS, Vancouver, BC, CAN	
☐ 703-978-9754	BBS, Annandale, VA	
☐ 602-246-1432	BBS Apollo, Phoenix, AZ	24h
☐ 816-587-9543	BBS Atari Amis, Kansas City, MO	24h
☐ 213-394-5950	BBS B.R., Los Angeles, CA	24h
☐ 401-521-2626	BBS Colornet, Providence, RI	● ★
☐ 809-781-0350	BBS Commodore, San Juan, PR	●
☐ 216-757-3711	BBS Computer Applications Co., Poland, OH	
☐ 707-585-3586	BBS Express	
☐ 401-738-5152	BBS Heathkit Store, Warwick, RI	●
☐ 305-246-1111	BBS Homestead, FL	
☐ 404-252-4146	BBS IBM Hostcomm, Atlanta, GA	
☐ 703-978-9592	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 703-978-0921	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 703-591-5120	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 703-425-9452	BBS IBM Hostcomm, Fairfax, VA	24h
☐ 713-890-0310	BBS IBM Hostcomm, Houston, TX	24h
☐ 703-425-7229	BBS IBM Hostcomm, Springfield, VA	24h
☐ 416-499-7023	BBS IBM Hostcomm, Toronto, ON, CAN	24h Ⓟ
☐ 703-560-0979	BBS IBM PC, Annandale, VA	24h
☐ 404-294-6879	BBS IBM PC, Atlanta, GA	
☐ 404-252-9438	BBS IBM PC, Atlanta, GA	24h
☐ 301-937-4339	BBS IBM PC, Beltsville, MD	24h
☐ 301-460-0538	BBS IBM PC, Bethesda, MD	24h
☐ 704-365-4311	BBS IBM PC, Charlotte, NC	24h
☐ 617-353-9312	BBS IBM PC, Computer Society, Boston, MA	●
☐ 213-649-1489	BBS IBM PC, Culver City, CA	24h ★
☐ 703-680-5220	BBS IBM PC, Dale City, VA	24h
☐ 301-251-6293	BBS IBM PC, Gaithersburg, MD	24h
☐ 703-759-5049	BBS IBM PC, Great Falls, VA	24h ★
☐ 608-262-4939	BBS IBM PC, Madison, WI	24h
☐ 312-991-8887	BBS IBM PC, Niles, IL	24h
☐ 301-949-8848	BBS IBM PC, Rockville, MD	24h
☐ 703-560-7803	BBS IBM PC, Vienna, VA	24h
☐ 312-882-4227	BBS IBM PCmodem, Chicago, IL	24h ★
☐ 312-376-7598	BBS IBM PCmodem, Chicago, IL	24h
☐ 713-661-5428	BBS MCUA, Houston, TX	24h
☐ 904-477-8783	BBS, Pensacola, FL	
☐ 414-483-4578	BBS SUE, Milwaukee, WI	
☐ 401-272-1138	BBS Syslink, Providence, RI	24h
☐ 612-724-7066	BBS The Safehouse, Minneapolis, MN	24h
☐ 707-527-5908	BBS-16, Santa Rosa, CA	
☐ 214-289-1386	BBS-80 Daltrug, Dallas, TX	24h
☐ 904-932-8271	Beach Game System	
☐ 414-259-9475	Big Top Games System, Milwaukee, WI	
☐ 408-267-7399	Bird House, San Jose, CA	
☐ 602-952-1382	Blax-80 BBS, Phoenix, AZ	24h
☐ 305-392-5927	Boca Harbor	
☐ 617-423-6985	Boston Information Exchange, Boston, MA	24h ★
☐ 416-487-5833	Bradley Brothers BBS, Toronto, ON, CAN	24h $
☐ 416-481-9047	Bradley Brothers BBS Download, Toronto, ON, CAN	24h $
☐ 813-734-7103	Bradley Computer BBS	
☐ 212-933-9459	Bronx BBS, New York, NY	
☐ 813-885-6187	BSBB, Tampa, FL	
☐ 408-980-0276	Buccaneer's Harbor	
☐ 416-265-3227	Bull 80, Toronto, ON, CAN	7:30pm-8am, 24h wknds
☐ 416-423-3265	Bull BBS (ETI Magazine), Toronto, ON, CAN	⚤
☐ 617-266-7789	Bullet-80, Boston, MA	24h ★
☐ 216-729-2769	Bullet-80, Chesterland, OH	
☐ 717-586-2112	Bullet-80, Clarks Summit, PA	
☐ 203-744-4644	Bullet-80, Danbury, CT	
☐ 915-565-9903	Bullet-80, El Paso, TX	24h
☐ 404-461-9686	Bullet-80, Fayetteville, GA	
☐ 205-492-0373	Bullet-80, Gadsden, AL	24h
☐ 601-264-2361	Bullet-80, Hattiesburg, MS	24h
☐ 712-368-2651	Bullet-80, Holstein, IA	
☐ 614-532-6920	Bullet-80, Ironton, OH	
☐ 215-364-2180	Bullet-80, Langhorne, PA	
☐ 212-740-5680	Bullet-80, New York, NY	24h
☐ 714-952-2110	Bullet-80, Orange County, Anaheim, CA	
☐ 714-644-7942	Bullet-80 Pirate Place	
☐ 203-888-7952	Bullet-80, Seymour, CT	
☐ 217-529-1113	Bullet-80, Springfield, IL	
☐ 313-683-5076	Bullet-80, Waterford, MI	24h
☐ 707-539-6471	Byte The Bulletin	

C

☐ 305-432-5969	Cable Box	
☐ 206-524-0203	Call-A.P.P.L.E., Seattle, WA	
☐ 602-275-6644	Call-A-Lawyer, Phoenix, AZ	24h
☐ 518-346-3596	Capital City BBS, Albany, NY	24h
☐ 617-279-0522	Captain Flint's Quarterdeck	
☐ 612-377-7747	Captain's Log	
☐ 703-823-5210	Carrier 2, Alexandria, VA	
☐ 312-598-4861	Cass-80, Hickory Hills, IL	
☐ 703-734-1387	CBBS Amrad, Washington, DC	24h
☐ 404-394-4220	CBBS, Atlanta, GA	24h
☐ 312-897-9037	CBBS Aurora Computer Peripherals, Aurora, CO	24h
☐ 504-273-3116	CBBS, Baton Rouge, LA	24h
☐ 812-334-2522	CBBS, Bloomington, IN	
☐ 617-646-3610	CBBS, Boston, MA	24h
☐ 319-364-0811	CBBS, Cedar Rapids, IA	24h
☐ 312-545-8086	CBBS, Chicago, IL	24h
☐ 301-948-5717	CBBS CPEUG/ICST, Gaithersburg, MD	
☐ 415-658-2919	CBBS Lambda, Berkeley, CA	⚤
☐ 617-683-2119	CBBS Lawrence General Hospital, Boston, MA	

516-561-6590	CBBS Lica Limbs, Long Island, NY	24h
4-1 399-2136	CBBS, London, England	
516-334-3134	CBBS, Long Island, NY	24h
414-241-8364	CBBS MAUDE, Milwaukee, WI	24h
617-752-7284	CBBS Microstar, Worcester, MA	
613-236-3009	CBBS Ottawa, ON, CAN	
503-646-5510	CBBS Portland, OR	24h
412-822-7176	CBBS PACC, Pittsburgh, PA	24h
604-562-9515	CBBS, Prince George, BC, CAN	
415-357-1130	CBBS Proxima, Berkeley, CA	
716-244-9531	CBBS Rams, Rochester, NY	
612-423-5016	CBBS, Rosemont, MN	
813-866-9945	CBBS, St. Petersburg, FL	24h
808-944-0562	CBBS Strictly Software, Honolulu, HI	
416-461-2110	CBBS, Toronto, ON, CAN	24h
602-746-3956	CBBS TSG, Tucson, AZ	24h
604-687-2640	CBBS Vancouver, BC, CAN	24h
312-259-8086	CBBS Ward And Randy's, Chicago, IL	
301-640-0498	Centaur Island	
304-925-3338	Century 21st	
416-366-2069	CFTR BBS, Toronto, ON, CAN	6pm-9am
314-434-6187	Chambers of Xenobia	
303-698-7620	Chess Board, Denver, CO	
303-753-1554	Cheyenne Mountain, Denver, CO	
415-820-0711	Chthon	
703-360-3812	C-HUG Bulletin Board, Fairfax, VA	24h
213-930-2578	CIA	
815-397-4176	Cider City	
312-957-3924	C.M.M.S., Chicago, IL	24h
414-476-8722	Coco-Mug	24h
416-743-6221	Coco-Nut, Toronto, ON, CAN	24h
518-235-9073	Cohoes Forum, Cohoes, NY	
213-336-5535	Coin Games Net	
414-543-3333	Color-80	24h
305-969-0000	Color Dimension 300, West Palm Beach, FL	
904-264-0335	Colour-80, Orange Park, FL	24h
416-767-0412	Colour-80, Toronto, ON, CAN	6pm-9am
212-897-3392	Comm-80, Queens, NY	24h
416-723-6500	Commodore 64 BBS, Oshawa, ON, CAN	
314-625-4576	Commodore Communication, St. Louis, MO	24h
414-679-9103	Commodore Up/Dnload Line..3pm-10pm	
312-674-6502	Commodore Video King, IL	
314-638-0644	Communitree Golden Hind, St. Louis, MO	24h
216-645-0827	Comnet-80, Akron, OH	24h ★
714-770-5052	Comnet-80, Laguna Hills, CA	
702-870-9986	Comnet-80, Las Vegas, NV	★
313-465-9531	Comnet-80, Mt. Clemens, MI	★
215-855-3809	Comnet-80, North Wales, PA	
714-359-3189	Comnet-80, Riverside, CA	★
714-877-2253	Comnet-80, Riverside, CA	★
817-767-5847	Comnet-80, Wichita Falls, TX	
516-775-5700	Compost	
713-444-7041	Compuque-80, Houston, TX	24h ★
803-771-0922	Compusystems, Columbia, SC	
301-587-2132	Computer Age Inc	
416-683-2226	Computer Camp BBS	5pm-9am
213-657-1799	Computer Connection, Los Angeles, CA	
805-496-0850	Computer Connection	
414-255-1222	Computer Palace, Milwaukee, WI	10am-10pm wknds
714-983-9923	Computers For Christ, Ontario, CA	24h
416-633-0185	Comspec BBS, Downsview, ON, CAN	
602-931-1829	Conference-Tree, Phoenix, AZ	24h
907-344-5251	Conference-Tree, Anchorage, AK	
404-982-9627	Conference-Tree, Atlanta, GA	24h
408-475-7101	Conference-Tree, Berkeley, CA	
808-487-2001	Conference-Tree Computerland, Honolulu, HI	24h
201-627-5151	Conference-Tree Flagship, Rockaway, NJ	24h
415-538-3580	Conference-Tree, Hayward, CA	
213-372-4800	Conference-Tree Kelp Bed, Los Angeles, CA	
612-854-9691	Conference-Tree, Minneapolis, MN	
415-861-6489	Conference-Tree, San Francisco, CA	
415-626-9427	Conference-Tree, San Francisco, CA	
213-394-1505	Conference-Tree, Santa Monica, CA	
415-332-8115	Conference-Tree, Sausalito, CA	
512-578-5833	Conference-Tree, Victoria, TX	
516-588-5836	Connection-80, Centereach, NY	
303-690-4566	Connection-80, Denver, CO	24h
415-651-4147	Connection-80, Fremont, CA	24h
301-840-8588	Connection-80, Gaithersburg, MD	24h
516-482-8491	Connection-80, Great Neck, NY	24h
904-353-5227	Connection-80 Jacs, Jacksonville, FL	24h
517-339-3367	Connection-80, Lansing, MI	
514-622-1274	Connection-80, Laval Bele, Laval, PQ, CAN	24h
212-991-1664	Connection-80, Manhattan, NY	
305-644-8327	Connection-80, Orlando, FL	24h
603-924-7920	Connection-80, Peterborough, NH	
813-977-0989	Connection-80, Tampa, FL	
616-457-1840	Connection-80 W. Mich. Micro Group, MI	24h
305-894-1886	Connection-80, Winter Garden, FL	24h
212-441-3755	Connection-80, Woodhaven, NY	24h
513-871-8901	Cook's Galley	
305-391-3893	C.O.P.S	
313-547-7903	CPU	
602-956-5021	Creepy Corridors, Phoenix, AZ	●
313-856-3804	Crystal Castle	
602-861-4090	Crystal, Phoenix, AZ	↔
619-691-8367	CVBBS, San Diego, CA	24h
713-376-6382	Cyrus Dimension	

D

213-633-5463	Data-Mate Canoga Park, CA	⚤
215-563-9815	Datanet 1200 Baud	
215-563-9211	Datanet 300 Baud	
414-672-6053	DataTech	24h
415-522-1986	Dataworx	
313-764-1837	Davy Jones Locker	
617-865-1264	Davy Jones Locker, Lexington, MA	
213-346-1849	Dec-Line, Woodland Hills, CA	24h ↔
612-938-7535	Deep Thot	
414-421-2863	Demon's Realm	6pm-6am
213-842-3322	Dial-Your-Match #1	⚤
619-434-4600	Dial-Your-Match #11, Carlsbad, CA	24h ⚤
713-556-1531	Dial-Your-Match #12, Houston, TX	24h ⚤
201-272-3686	Dial-Your-Match #14, Cranford, NJ	⚤
206-256-6624	Dial-Your-Match #16, Seattle, WA	⚤
415-991-4911	Dial-Your-Match #17	⚤
617-334-6369	Dial-Your-Match #18	⚤
919-362-0676	Dial-Your-Match #20	⚤
201-462-0435	Dial-Your-Match #21, Freehold, NJ	⚤
213-990-6830	Dial-Your-Match #22	⚤
402-571-8942	Dial-Your-Match #23, Omaha, NE	⚤
713-783-4136	Dial-Your-Match #24, Houston, TX	⚤
209-298-1328	Dial-Your-Match #26, Clovis, CA	⚤
912-233-0863	Dial-Your-Match #3	⚤
619-748-8746	Dial-Your-Match #33, Poway, CA	24h ⚤
312-243-1046	Dial-Your-Match #39, Chicago, IL	⚤
213-783-2305	Dial-Your-Match #4	⚤
415-467-2588	Dial-Your-Match #8, San Francisco, CA	⚤
213-345-1047	Dial-Your-Match #9	⚤
212-541-5975	Dial-Your-Match, New York, NY	⚤
602-890-0972	Diamond III, Phoenix, AZ	24h
714-974-9788	Dimension-80, Orange, CA	
514-327-5764	Distra-Soft, Montreal, PQ, CAN	24h
713-471-4131	Doc Board, Houston, TX	
301-926-3470	Doctor's Office	
415-488-9145	Download-80 Mojo's, Forest Knolls, CA	24h ★
213-347-9780	Dr. Falcon's Retreat, Canoga Park, CA	★
416-421-8930	Dr. Phobos Dating BBS, Toronto, ON, CAN	24h
817-665-3876	Dragonfire	
213-428-5206	Dragons Game System	Ⓟ = dragon
414-282-0501	Dragons Lair, Milwaukee, WI	
408-996-7464	Dragons Lair	
415-552-7671	Drummer	⚤
215-855-3809	Dru's Communique-80	
707-527-5908	Dual BBS 16	
714-841-5321	Dune	
313-644-3841	DWBBS	Ⓟ = BBS, UN = DW.BBS

E

213-789-9512	Electric Line Connection, Sherman Oaks, CA	
212-997-2488	Electronic Bookshelf	
313-474-5795	Electronic Odyssey	
314-645-1047	EMC-80, St. Louis, MO	
414-835-1754	E.S.C.A.P.E	Ⓟ
613-236-3009	ETW BBS, Ottawa, ON, CAN	
416-921-4013	Exceltronics, Toronto, ON, CAN	24h
414-964-5160	Exec-PC	24h
913-676-3613	Experimental-80, Kansas City, MO	

F

314-991-2744	Fantasy Island	
213-840-8066	Fantasy Plaza	
713-530-5249	Fantasy Voyage	
317-494-6643	FBBS #1, Purdue, IN	24h ★
714-532-4521	Flipper's, Garden Grove, CA	
815-455-2406	Flynn's Games	
303-465-2027	Forbidden Zone	
303-399-8858	Forum-80 #2, Denver, CO	24h
404-279-5392	Forum-80, Augusta, GA	
803-552-1612	Forum-80, Charleston, SC	24h
216-486-4176	Forum-80, Cleveland, OH	★
915-755-1000	Forum-80, El Paso, TX	24h
305-772-4444	Forum-80, Ft. Lauderdale, FL	24h
44 482859169	Forum-80, Hull, England	(Country Code = 011)
816-861-7040	Forum-80, Kansas City, MO	24h ★
816-931-9316	Forum-80, Kansas City, MO	★
702-362-3609	Forum-80, Las Vegas, NV	24h
201-486-2956	Forum-80, Linden, NJ	24h
503-535-6883	Forum-80, Medford, OR	24h
901-276-8196	Forum-80 Medical, Memphis, TN	24h
201-528-6623	Forum-80 Monmouth, Brielle, NJ	24h
205-272-5069	Forum-80, Montgomery, AL	
603-882-5041	Forum-80, Nashua, NH	
613-820-4646	Forum-80, Ottawa, ON, CAN	
703-670-5881	Forum-80, Prince William County, VA	24h
415-348-2139	Forum-80, San Mateo, CA	★
206-723-3282	Forum-80, Seattle, WA	
602-458-3850	Forum-80, Sierra Vista, AZ	24h
617-692-3973	Forum-80, Westford, MA	
316-682-2113	Forum-80, Wichita, KS	24h ★
503-635-7205	Freebooter's Archives	
703-360-5439	Future Tech, Alexandria, VA	24h

G

713-444-7098	GABBS Armadillo Media, Houston, TX	24h
713-455-6502	GABBS, Houston, TX	24h
602-991-0144	Garden Of Eden, Phoenix, AZ	24h
301-344-9156	Gas Net	
416-439-0065	Games BBS, Scarborough, ON, CAN	7pm-9am
303-693-1064	GBBSII, Denver, CO	●
303-469-7541	GBBSII Apple Pi, CO	24h
303-343-8401	GBBSII Aurora-Net, Denver, CO	24h
303-750-3783	GBBSII Eamon, Denver, CO	● ★
303-443-3367	GBBSII Off The Wall, Denver, CO	24h
414-282-4181	Generic, Milwaukee, WI	Ⓟ
602-967-4529	Genesys, Phoenix, AZ	24h
416-482-2823	G.E. Nightowl, Toronto, ON, CAN	24h
216-845-3179	Genius' Modemline	
416-877-0933	Georgetown HAM Radio BBS, Georgetown, ON, CAN	
707-538-9124	Grape Vine BBS, Napa Valley, CA	24h
312-622-4442	Greene Machine, Chicago, IL	⚤
305-968-8653	Greene Machine Corsair, West Palm Beach, FL	
213-445-3591	Greene Machine Fricaseed Chicken, Arcadia, CA	24h

☐ 415-897-2783	Greene Machine Golden State BBS, Novato, CA	
☐ 213-431-1443	Greene Machine, Los Alamitos, CA	
☐ 714-354-8004	Greene Machine, Riverside, CA	
☐ 315-337-7720	Greene Machine, Rome, NY	
☐ 213-287-1363	Greene Machine, Temple City, CA	
☐ 305-965-4388	Greene Machine, West Palm Beach, FL.	⚤
☐ 602-726-7533	Greene Machine, Yuma, AZ	24h ★
☐ 213-591-7239	Groundstar System, Long Beach, CA	24h
	H	
☐ 217-877-1544	Hacker's Haven	
☐ 301-593-7033	Handicapped Exchange	
☐ 617-332-5017	Hanger 19	
☐ 516-328-8204	Hardware Haven	
☐ 516-367-8172	Haunted Mansion	
☐ 414-255-9645	H.A.U.S.E, Milwaukee, WI	7pm-7am
☐ 616-531-0890	HBBS Heath/Zenith, Grand Rapids, MI	★
☐ 213-366-1238	HBBS Mog-ur, Granada Hills, CA	24h ★
☐ 604-430-8233	Heath BBS, Vancouver, BC, CAN	
☐ 215-434-3998	Hermes-80, Allentown, PA	
☐ 301-593-7033	Hex, Silver Spring, MD	24h
☐ 415-674-0660	Human & Wisdom	
	I	
☐ 415-481-0252	IBM PC No-name, San Lorenzo, CA	24h ★
☐ 714-545-7359	IDBN Info-Net, Costa Mesa, CA	
☐ 216-724-2125	Infoex-80, Akron, OH	24h
☐ 918-838-8698	Infoex-80, Tulsa, OK	24h
☐ 305-683-6044	Infoex-80, West Palm Beach, FL	24h
☐ 416-278-3267	Infoport, Port Credit, ON, CAN	24h
☐ 416-762-1820	Insane Asylum, Toronto, ON, CAN	10pm-8am
☐ 213-477-4605	Interface, Los Angeles, CA	
☐ 312-296-3883	Interface BBS (Atari), Chicago, IL	
☐ 714-551-4336	Irvine Line, Irvine, CA	
	J	
☐ 206-883-0403	JCTS, Redmond, WA	24h
☐ 713-932-1124	Jolly Roger #2, Houston, TX	
	K	
☐ 206-767-7777	Kingdom of Seven, Seattle, WA	
☐ 615-297-6037	Knight Line	
☐ 212-631-1788	Kracker's Kastle	
☐ 213-947-8128	Kluge Computer	24h ★
	L	
☐ 213-631-3186	L.A. Interchange, Los Angeles, CA	24h
☐ 303-423-3156	Laboratory I	
☐ 303-751-2063	Laboratory II (Land of Oz), Denver, CO	
☐ 815-397-4176	Laboratory III	
☐ 215-435-3388	Lehigh Press BBS, Allentown, PA	
☐ 403-320-6923	Lethbridge Gaming System, Lethbridge, AB	
☐ 318-237-3350	Linc	
☐ 415-522-6441	Litterbox	
☐ 415-565-3037	Living BBS, Education SIG	
☐ 416-445-5192	Logic BBS, North York, ON, CAN	24h $
	M	
☐ 213-470-5912	Mad Board From Mars, Los Angeles, CA	
☐ 402-734-4748	Mages Inn, Omaha, NE	24h
☐ 703-471-0310	Magus	
☐ 703-471-0611	Magus, Herndon, VA	24h
☐ 318-989-8537	Magic Kingdom	
☐ 602-251-8538	Magic Lantern	
☐ 303-694-2871	Magic Window, Denver, CO	
☐ 206-527-0897	Mail Board-82, Seattle, WA	24h
☐ 303-986-5039	Mansion, Denver, CO	
☐ 414-224-6930	Marquette	Ⓟ
☐ 312-674-9246	Marvin	
☐ 213-478-5478	Master World, Los Angeles, CA	
☐ 414-241-8364	M.A.U.D.E	24h
☐ 312-927-1020	MCMS C.A.M.S. Chicago, IL	24h ★
☐ 612-753-3082	MCMS Goliath, Minneapolis, MN	
☐ 815-838-1020	MCMS J.A.M.S. Lockport, IL	24h
☐ 312-260-0640	MCMS Metro West Database, Chicago, IL	24h ★
☐ 612-533-1957	MCMS NC Software, Minneapolis, MN	24h
☐ 312-462-7560	MCMS P.C.M.S. Wheaton, IL	24h ★
☐ 312-351-4374	MCMS Waco Hot Line, Schaumburg, IL	24h Ⓟ
☐ 217-753-4309	MCMS Word Exchange, Springfield, IL	24h
☐ 416-978-6893	Medical Net-Works, Toronto, ON, CAN	7pm-9am
☐ 604-591-6975	Message 80, Surrey, BC, CAN	24h
☐ 416-782-9686	Micro 80, Toronto, ON, CAN	8pm-8am
☐ 305-686-3695	Micro-80, West Palm Beach, FL	
☐ 216-875-4582	Micro-COM, Louisville, OH	24h
☐ 301-560-9555	Micro Encounter	
☐ 813-875-3331	Micro Informer, Tampa, FL	
☐ 504-631-3589	Micro Phone	
☐ 604-224-2337	Microstat, BC, CAN	
☐ 602-938-4508	MicroSystems, Phoenix, AZ	24h
☐ 414-353-2402	Midnight Star	10pm-1pm
☐ 314-227-4312	Midwest, St. Louis, MO	⚤
☐ 312-279-4399	Midwest Pirate System	
☐ 414-377-3878	Midwest Software Library..5pm-6am	
☐ 414-327-5300	Milwaukee Express, Milwaukee, WI	24h $
☐ 414-281-0545	Milwaukee Tribune, Milwaukee, WI	24h
☐ 713-871-8577	Mines of Moria	
☐ 408-688-9629	Mines of Moria II, Aptos, CA	
☐ 206-762-5141	Mini-Bin, Seattle, WA	24h
☐ 414-774-8478	Mini-Board	wknds
☐ 203-744-4644	Mini-Serve	
☐ 301-983-8293	Mission Control	
	MMMMM - MARC The Martian's Mixed Up Matching Machine	
☐ 213-390-3239	MMMMM#1, Santa Monica, CA..(line One)	★ ⚤
☐ 213-450-4580	MMMMM#1, Santa Monica, CA..(line Two)	⚤
☐ 212-541-5975	MMMMM#2, New York, NY	⚤
☐ 213-452-6111	MMMMM#3, Marina Del Rey, CA	⚤
☐ 213-821-2257	MMMMM#4, Lawndale, CA	⚤
☐ 305-755-5560	Mordor	

☐ 312-759-9191	Mother	
☐ 313-453-5146	Motherboard	
☐ 415-352-8442	Motherboard, San Leandro, CA	
☐ 416-728-6574	Motor City BBS, Oshawa, ON, CAN	
☐ 206-334-7394	MSG-80, Everett, WA	
☐ 309-797-8535	Mystery Castle	
	N	
☐ 804-444-3392	NBBS, Norfolk, VA	
☐ 812-858-5405	Net-Works II	
☐ 816-483-2526	Net-Works ABC, Kansas City, MO	
☐ 318-988-1302	Net-Works Acadiana, LA	
☐ 312-295-7284	Net-Works Adventure's Inn, Lake Forest, IL	24h
☐ 404-733-3461	Net-Works AGS, Augusta, GA	24h
☐ 512-623-6123	Net-Works Alamo City, TX	
☐ 907-278-4223	Net-Works Alaska	
☐ 305-772-1076	Net-Works Apple Barrel, FL	
☐ 415-585-6334	Net-Works Apple Corps, San Francisco, CA	
☐ 318-861-1012	Net-Works Apple Gumbo, Shreveport, LA	24h
☐ 714-823-1451	Net-Works Apple Jacks, CA	
☐ 312-685-9573	Net-Works Apple Juice, Drien, IL	
☐ 312-963-5384	Net-Works Apple Net, Chicago, IL	
☐ 409-846-2900	Net-Works Apple Seed, College Station, TX	24h
☐ 214-644-4781	Net-Works Apple Shack, TX	
☐ 312-935-3091	Net-Works Apple-Technical, Chicago, IL	
☐ 701-746-4959	Net-Works Armadillo, Grand Forks, ND	
☐ 502-459-5531	Net-Works Assembly Line, Louisville, KY	●
☐ 618-692-0742	Net-Works Asylum, IL	
☐ 502-423-0695	Net-Works Baud-Ville, Louisville, KY	●
☐ 904-932-8271	Net-Works Beach BBS, Pensacola, FL	
☐ 305-948-8000	Net-Works Big Apple, Miami, FL	
☐ 713-782-5706	Net-Works Briar-Net, Houston, TX	24h
☐ 212-410-0949	Net-Works, Brooklyn, NY	
☐ 217-429-4738	Net-Works C.A.M.S., Decatur, IL	24h
☐ 304-345-8280	Net-Works, Charleston, WV	
☐ 312-882-9237	Net-Works Chicago, IL	
☐ 312-323-3741	Net-Works Chipmunk, Hinsdale, IL	24h
☐ 312-255-6489	Net-Works CLAH, Chicago, IL	
☐ 213-336-5535	Net-Works Coin Games, Los Angeles, CA	
☐ 301-953-3341	Net-Works Comm Center NW3NAGAD, Laurel, MD	
☐ 817-261-4700	Net-Works Compushop FWA, TX	
☐ 401-331-8450	Net-Works Computer City, RI	
☐ 408-227-5416	Net-Works Computer Emporium, CA	
☐ 515-279-8863	Net-Works Computer Emporium, IA	
☐ 301-543-9429	Net-Works Computer Island, MD	
☐ 808-524-6668	Net-Works Computer Market, Honolulu, HI	●
☐ 817-732-1787	Net-Works Computer Pro, Ft. Worth, TX	
☐ 314-432-7120	Net-Works Computer Station, MO	
☐ 808-488-7756	Net-Works Computer Store, Honolulu, HI	
☐ 213-859-0894	Net-Works Computer World, Los Angeles, CA	24h
☐ 504-454-6688	Net-Works Crescent City, LA	
☐ 214-361-1386	Net-Works, Dallas, TX	
☐ 513-223-3672	Net-Works, Dayton, OH	
☐ 312-627-5138	Net-Works Death Star, Oakbrook, IL	24h
☐ 214-239-5842	Net-Works Eclectic Computer Sys., Dallas, TX	
☐ 915-593-6655	Net-Works El Paso, TX	
☐ 315-768-8153	Net-Works Elppa System, NY	
☐ 213-345-3670	Net-Works Encino, CA	
☐ 314-532-4652	Net-Works Forth Dimension, St. Louis, MO	
☐ 215-244-0864	Net-Works Galaxy One, PA	
☐ 313-455-4227	Net-Works GBBS Metro Detroit, MI	⚤
☐ 618-877-2904	Net-Works, Granite City, IL	
☐ 317-326-3833	Net-Works, Greenfield, IN	24h
☐ 618-254-6074	Net-Works Harpos Bar & Grill, IL	
☐ 808-423-1593	Net-Works Hawaii Connection, Honolulu, HI	24h
☐ 808-521-7312	Net-Works Hawaii, Honolulu, HI	
☐ 314-968-7225	Net-Works Infoline, MO	
☐ 713-468-0174	Net-Works Jolly Roger, Houston, TX	24h
☐ 414-727-3637	Net-Works Lab-Works, WI	
☐ 913-648-6071	Net-Works Leawood, KS	
☐ 201-994-9620	Net-Works, Livingston, NJ	24h
☐ 309-342-7178	Net-Works Magie, Galesburg, IL	
☐ 213-388-5198	Net-Works Magnetic Fantasies, Los Angeles, CA	
☐ 617-256-1446	Net-Works Micro BBS, Chelmsford, MA	
☐ 713-864-4672	Net-Works Micro Design, Houston, TX	●
☐ 312-998-5066	Net-Works Micro Ideas, Glenview, IL	
☐ 707-528-3462	Net-Works Micro-Sys, CA	
☐ 713-871-8577	Net-Works Mines Of Moria, Houston, TX	24h
☐ 618-466-9497	Net-Works NAGS, IL	
☐ 812-858-5405	Net-Works Nick Naimo, Newburgh, IN	
☐ 503-655-6009	Net-Works Oregon City, OR	
☐ 617-494-1985	Net-Works Pirate's Harbor, MA	
☐ 617-720-3600	Net-Works Pirate's Harbor, Boston, MA	
☐ 213-454-3075	Net-Works Pirate's Inn, CA	
☐ 914-634-1268	Net-Works Pirate's Lodge, NY	
☐ 713-974-5258	Net-Works Pirate's Palace, Houston, TX	24h
☐ 312-935-2933	Net-Works Pirate's Ship, IL	
☐ 516-627-9048	Net-Works Pirate's Trek	
☐ 603-436-3461	Net-Works, Portsmouth, NH	
☐ 312-393-4755	Net-Works RJNET, Warnville, IL	
☐ 213-473-2754	Net-Works Softworx, West Los Angeles, CA	
☐ 314-821-5826	Net-Works Space Age, MO	
☐ 314-994-9257	Net-Works St. Louis Exchange, MO	
☐ 713-333-2309	Net-Works The Dark Realm, Houston, TX	24h
☐ 408-996-7464	Net-Works The Dragon's Lair NW,	
☐ 713-354-4690	Net-Works The Inner Realm, Houston, TX	24h
☐ 713-777-8608	Net-Works The Shadow World, Houston, TX	24h
☐ 816-232-3153	Net-Works The Silver Tongue, St. Joseph, MO	
☐ 713-785-7996	Net-Works The System, Houston, TX	●
☐ 713-492-8700	Net-Works The Weekender, Houston, TX	24h
☐ 416-683-3733	Net-Works, Toronto, ON, CAN	24h ★
☐ 416-445-6696	Net-Works, Toronto, ON, CAN	24h
☐ 805-522-1789	Net-Works Visual Comm, CA	
☐ 317-743-8667	Net-Works Von's Electronics, IL	
☐ 618-345-6638	Net-Works Warlock's Castle St. Louis, MO	
☐ 214-824-7455	Net-Works Winesap, TX	

☐ 713-933-7353	Net-Works Zachary-Net, Houston, TX	24h
☐ 303-985-9184	Neutral Zone, Denver, CO	
☐ 518-370-8343	Nibble One, Schenectady, NY	
☐ 415-482-2823	Night Owl	
☐ 714-633-5240	Nortec BBS, Toronto, ON, CAN	24h
☐ 714-633-5240	North Orange County Computer Club, Orange, CA	
☐ 218-727-2184	Northeast Minnesota Net	
☐ 305-686-4862	Notebook, West Palm Beach, FL	
☐ 213-881-6880	Novation Co., Los Angeles, CA	℗ = cat
☐ 202-363-8165	NWDS	
☐ 318-688-7078	NWLAIBMPCUG, Shreveport, LA	
☐ 206-743-6021	NWWCUG Edmunds, Seattle, WA	
☐ 914-592-5385	Nybbles-80, Elmsford, NY	
☐ 212-626-0375	Nybbles-80, New York, NY	

O

☐ 402-292-9598	OACPM, Omaha, NE	24h
☐ 503-641-2798	OARCS, Portland, OR	
☐ 714-530-8226	OCTUG Orange County, Garden Grove, CA	
☐ 303-443-3367	Off The Wall	
☐ 614-423-4422	Ohio Valley BBS	
☐ 602-952-2018	Omega, Phoenix, AZ	24h
☐ 514-931-0458	Online Computerland, Montreal, PQ, CAN	24h
☐ 913-432-5544	Online Dickinsons Movie Guide, Mission, KS	24h
☐ 317-787-9881	Online, Indianapolis, IN	24h ℗ = pass, id# = gues
☐ 312-648-4867	Online Omega, Chicago, IL	24h
☐ 619-692-1961	Online Saba, San Diego, CA	24h
☐ 612-546-1013	On-Target	
☐ 213-980-5643	Oracle, North Hollywood, CA	℘
☐ 714-537-7913	Orange County Data Exchange, Garden Grove, CA	
☐ 312-397-8308	OS-9 6809 BBS, Palatine	
☐ 416-484-9663	OSBOARD, Toronto, ON, CAN	24h
☐ 914-725-4060	OSUNY, Scarsdale, NY	
☐ 213-784-0204	Outer Limits # 1, Van Nuys, CA	24h
☐ 213-782-8390	Outer Limits # 2, Van Nuys, CA	
☐ 312-441-6957	Outpost	

P

☐ 604-584-1047	Pacific Blue, BC, CAN	
☐ 501-372-0576	PBBS Arc-Net, Little Rock, AR	24h
☐ 312-359-9450	PBBS Co-operative Comp SVC, Palatine, IL	24h
☐ 619-561-7271	P.DBMS Lakeside, CA	24h ★
☐ 205-972-1685	Pentagon	
☐ 305-427-6300	Personal Msg. System-80, Deerfield Beach, FL	24h ★
☐ 317-255-5435	PET BBS AVC Comline, Indianapolis, IN	24h
☐ 312-397-0871	PET BBS Commodore, Chicago, IL	24h
☐ 813-391-5219	PET BBS Commodore, Largo, FL	
☐ 416-624-5431	PET BBS PSI Wordpro, Mississauga, ON, CAN	24h
☐ 414-554-9520	PET BBS S.E.W.P.U.G., Racine, WI	24h
☐ 307-637-6045	PET BBS SE Wyoming PUG	24h
☐ 416-782-9534	PET BBS TPUG, Toronto, ON, CAN	24h ℗
☐ 309-729-9518	Phantom's Mansion	
☐ 213-360-0211	Phantoms Hollow Granada Hills, CA	
☐ 201-790-6795	Photo-80, Haledon, NJ	
☐ 714-545-8100	Pig Sty, Costa Mesa, CA	
☐ 304-744-2253	Pirate-80	
☐ 415-775-2384	Pirates Bay	
☐ 514-332-3443	Pirates Brigade, Montreal, PQ, CAN	
☐ 617-891-1349	Pirates Chest	
☐ 516-698-4008	Pirates Cove	
☐ 201-736-4630	Pirates Distributing	
☐ 314-576-4109	Pirates Emporium	
☐ 314-991-2744	Pirates Forge	
☐ 617-863-1237	Pirates Hideout, Lexington, MA	
☐ 201-366-2209	Pirates I/O	
☐ 612-825-5852	Pirates Island	
☐ 301-869-8747	Pirates Landing	
☐ 914-634-1268	Pirates Lodge	
☐ 305-335-8640	Pirates Loft II	
☐ 213-472-4287	Pirates Mountain, Los Angeles, CA	
☐ 206-783-9798	Pirates Of Puget Sound, Seattle, WA	
☐ 213-395-9813	Pirates Paper, Santa Monica, CA	
☐ 805-492-3150	Pirates Phunhouse, Thousand Oaks, CA	
☐ 313-968-2645	Pirates Prison II	
☐ 305-823-2756	Pirates Reef II	
☐ 305-854-6398	Pirates Reef	
☐ 703-644-1665	Pirates Trove	
☐ 703-323-4791	Pirates Trove III	
☐ 415-924-6282	Pirates Warehouse	
☐ 201-423-0810	Places Unknown	
☐ 516-935-2481	Plover Net	
☐ 713-441-4032	PMBBS	
☐ 714-772-8868	PMS " "if" ", Anaheim, CA	24h
☐ 907-344-8558	PMS, Anchorage, AK	
☐ 816-252-0232	PMS Apple Bits, Kansas City, MO	24h
☐ 617-767-1303	PMS Apple Guild, Weymouth, MA	24h
☐ 301-764-1995	PMS, Baltimore, MD	24h
☐ 702-878-9106	PMS Century 23, Las Vegas, NV	24h
☐ 312-373-8057	PMS, Chicago, IL	24h
☐ 513-671-2753	PMS, Cincinnati, OH	
☐ 617-774-7516	PMS Computer City, Danvers, MA	
☐ 619-582-9557	PMS Computer Merchant, San Diego, CA	24h
☐ 503-689-2655	PMS Computer Solutions, Eugene, OR	24h
☐ 619-271-8613	PMS Datel Systems Inc., San Diego, CA	24h
☐ 312-964-6513	PMS Downers Grove/Srt, Downers Grove, IL	
☐ 619-265-3428	PMS Ed Tech, San Diego, CA	
☐ 301-465-3176	PMS, Ellicott City, MD	
☐ 619-746-0667	PMS, Escondido, CA	●
☐ 619-579-7036	PMS Floppy House, San Diego, CA	24h
☐ 619-251-8538	PMS Floppy House	
☐ 501-646-0197	PMS Ft. Smith Comp. Club, Ft. Smith, AK	
☐ 409-233-7943	PMS Gulfcoast, Freeport, TX	24h
☐ 312-295-6926	PMS I.A.C., Lake Forest, IL	24h
☐ 317-787-5486	PMS, Indianapolis, IN	24h
☐ 619-578-2646	PMS Kid's Message System, San Diego, CA	24h
☐ 416-445-5192	PMS Logic Inc., Toronto, ON, CAN	24h $
☐ 213-331-3574	PMS, Los Angeles, CA	24h
☐ 216-832-8392	PMS, Massillon, OH	24h
☐ 212-997-2488	PMS Mcgraw-Hill Books, New York, NY	
☐ 612-929-6699	PMS, Minneapolis, MN	24h
☐ 213-346-1849	PMS O.A.C., Woodland Hills, CA	24h
☐ 301-653-3413	PMS, Pikesville, MD	
☐ 415-462-7419	PMS, Pleasanton, CA	24h
☐ 503-245-2536	PMS, Portland, OR	24h
☐ 415-851-3453	PMS, Portola Valley, CA	24h
☐ 216-867-7463	PMS Raug, Akron, OH	24h
☐ 415-490-7878	PMS Redington Group, Fremont, CA	24h
☐ 201-932-3887	PMS Rutgers Univ. Microlab, Piscataway, NJ	
☐ 619-727-7500	PMS, San Marcos, CA	24h
☐ 408-688-9629	PMS Santa Cruz, Aptos, CA	24h
☐ 619-561-7277	PMS, Santee, CA	24h
☐ 904-743-7050	PMS SEB Computer, Jacksonville, FL	
☐ 206-486-2368	PMS Software Unlimited, Kenmore, WA	24h
☐ 612-929-8966	PMS Twin Cities, Minneapolis, MN	
☐ 913-677-1299	PMS Your Computer Connection, Kansas City, MO	●
☐ 301-356-5895	Possesion	
☐ 617-965-2436	Post Office	
☐ 703-379-0303	Potomac Micro Magic Inc., Falls Church, VA	24h
☐ 301-994-0399	Program Store BBS, Baltimore, MD	24h
☐ 202-337-4694	Program Store BBS, Washington, DC	24h
☐ 305-763-1654	Project Blue Book	
☐ 415-357-1130	Proxima CBBS	

R

☐ 914-942-2638	RACS III	
☐ 714-524-1228	RACS V, Fullerton, CA	
☐ 414-784-0830	Radio Free Milwaukee, Milwaukee, WI	24h
☐ 217-429-6310	Rag Time Phreak, Decatur, IL	
☐ 201-887-8874	RATS System, Whippany, NJ	
☐ 609-468-5293	RATS, Wenonah, NJ	
☐ 609-468-3844	RATS, Wenonah, NJ #2	
☐ 312-876-0974	RBBS Milwaukee-Chicago Line	
☐ 213-368-5801	RBBS, San Fernando, CA	
☐ 213-395-0460	RBBS, Santa Monica, CA	★
☐ 312-647-7636	RCP/M A.B. Dick Co., Niles, IL	24h ★
☐ 907-337-1984	RCP/M, Anchorage, AK	●
☐ 703-536-3769	RCP/M, Arlington, VA	●
☐ 619-256-3914	RCP/M, Barstow, CA	24h ★
☐ 503-641-7276	RCP/M, Beaverton, OR	24h
☐ 713-438-2247	RCP/M, Blue Ridge, Missouri City, TX	24h
☐ 303-499-9169	RCP/M, Boulder, CO	●
☐ 312-326-4392	RCP/M, Bridgeport, IL	24h
☐ 714-774-7860	RCP/M CBBS Anahug, Anaheim, CA	24h
☐ 614-272-2227	RCP/M CBBS, Columbus, OH	24h
☐ 805-527-9321	RCP/M CBBS CP/M Net Simi Valley, CA	
☐ 214-931-8274	RCP/M CBBS, Dallas, TX	●
☐ 604-937-0906	RCP/M CBBS Frog Hollow, Vancouver, BC, CAN	24h
☐ 214-241-1939	RCP/M CBBS Maxicom, Farmers Branch, TX	24h ★
☐ 214-247-5307	RCP/M CBBS Maxicom, Line 2	
☐ 613-762-5088	RCP/M CBBS Micom, Melbourne, VIC, Australia	24h
☐ 213-799-1632	RCP/M CBBS, Pasadena, CA	24h
☐ 703-524-2549	RCP/M CBBS RLP, Maclean, VA	24h
☐ 916-483-8718	RCP/M CBBS, Sacramento, CA	24h
☐ 313-846-6127	RCP/M CBBS Technical, Detroit, MI	24h ★
☐ 503-621-3193	RCP/M Chuck Forsberg, OR	24h ★
☐ 408-263-2588	RCP/M Colossal Oxgate, San Jose, CA	
☐ 814-238-4857	RCP/M Cug-Node, PA State College	24h
☐ 303-781-4937	RCP/M Cug-Note, Denver, CO	24h
☐ 403-454-6093	RCP/M Dave Mccrady, Edmonton, AB, CAN	24h ★
☐ 408-378-8733	RCP/M Dbase II, San Jose, CA	24h
☐ 313-584-1044	RCP/M, Detroit, MI	
☐ 312-972-6979	RCP/M El Division, Argonne, IL	
☐ 201-584-9227	RCP/M, Flanders, NJ	24h ★
☐ 309-944-5455	RCP/M, Geneseo, IL	
☐ 312-469-2597	RCP/M Glen Ellyn, Chicago, IL	24h
☐ 213-360-5053	RCP/M, Granada Hills, CA	24h
☐ 312-967-0052	RCP/M Ham Radio, Morton Grove, IL	
☐ 416-335-6620	RCP/M HAPN Hamilton, ON, CAN	24h
☐ 312-252-2136	RCP/M Logan Square, Chicago, IL	24h
☐ 213-296-5927	RCP/M, Los Angeles, CA	24h
☐ 313-759-6569	RCP/M MCBBS Keith Petersen, Royal Oak, MI	
☐ 516-751-5639	RCP/M Mid-Suffolk, Long Island, NY	●
☐ 913-362-9583	RCP/M, Mission, KS	24h ★
☐ 416-232-0442	RCP/M Mississauga HUG, Mississauga, ON, CAN	24h ★
☐ 312-949-6189	RCP/M NEI, Chicago, IL	● ★
☐ 312-937-5639	RCP/M North Chicago, Chicago, IL	
☐ 312-251-0168	RCP/M North Side BBS, Chicago, IL	
☐ 206-357-7400	RCP/M, Olympia, WA	24h
☐ 408-867-1243	RCP/M Oxgate 001, Saratoga, CA	24h ★
☐ 804-898-7493	RCP/M Oxgate 007, Grafton, VA	24h
☐ 409-845-0509	RCP/M Oxgate College Station, TX	24h
☐ 207-839-2337	RCP/M Programmers Anonymous, Gorham, ME	24h ★
☐ 401-751-5025	RCP/M Providence, Providence, RI	
☐ 312-789-0499	RCP/M RBBS Aims, Hinsdale, IL	24h
☐ 215-398-3937	RCP/M RBBS, Allentown, PA	24h
☐ 913-843-4259	RCP/M RBBS Alphanet, Lawrence, KS	●
☐ 303-634-1158	RCP/M RBBS Arvada Elect, Colorado Springs, CO	24h
☐ 301-229-3196	RCP/M RBBS, Bethesda, MD	
☐ 301-661-2175	RCP/M RBBS BHEC, Baltimore, MD	24h
☐ 914-279-5693	RCP/M RBBS, Brewster, NY	●
☐ 513-489-0149	RCP/M RBBS, Cincinnati, OH	●
☐ 915-533-2202	RCP/M RBBS Comp. Tech. Assoc., El Paso, TX	24h
☐ 403-482-6854	RCP/M RBBS Computron, Edmonton, AB, CAN	24h
☐ 201-272-1874	RCP/M RBBS, Cranford, NJ	24h
☐ 415-595-0541	RCP/M RBBS Datatech 001, San Carlos, CA	24h ★
☐ 408-238-9621	RCP/M RBBS Datatech 007, San Jose, CA	24h
☐ 408-732-9190	RCP/M RBBS Datatech 010, Sunnyvale, CA	
☐ 915-598-1668	RCP/M RBBS, El Paso, TX	24h ★
☐ 707-422-7256	RCP/M RBBS, Fairfield, CA	
☐ 803-548-0900	RCP/M RBBS, Fort Mill, SC	24h
☐ 714-534-1547	RCP/M RBBS GFRN Data Exchange, Garden Grove, CA	24h ★
☐ 213-541-2503	RCP/M RBBS GFRN Data Exchange, Palos Verdes, CA	24h ★
☐ 319-363-3314	RCP/M RBBS Hawkeye-PC, Cedar Rapids, IA	

☐ 406-443-2768	RCP/M RBBS Helena Valley, Helena, MT	
☐ 213-653-6398	RCP/M RBBS, Hollywood, CA	24h
☐ 213-973-2374	RCP/M RBBS IBM-PC, Hawthorne, CA	★
☐ 305-830-4340	RCP/M RBBS IBM-PC, Orlando, FL	24h ★
☐ 904-725-4995	RCP/M RBBS JUG, Jacksonville, FL	24h ★
☐ 303-985-1108	RCP/M RBBS Lakewood, Denver, CO	24h
☐ 415-461-7726	RCP/M RBBS, Larkspur, CA	24h
☐ 301-953-3753	RCP/M RBBS, Laurel, MD	24h
☐ 212-255-7240	RCP/M RBBS Manhattan, New York, NY	24h ★
☐ 415-383-0473	RCP/M RBBS Marin County, CA	24h
☐ 205-895-6749	RCP/M RBBS NACS/UAH, Huntsville, AL	24h
☐ 707-257-6502	RCP/M RBBS Napa Valley, CA	24h
☐ 201-775-8705	RCP/M RBBS, Ocean, NJ	★
☐ 305-671-2330	RCP/M RBBS, Orlando, FL	24h ★
☐ 213-577-9947	RCP/M RBBS, Pasadena, CA	24h ★
☐ 201-747-7301	RCP/M RBBS Paul Bogdanovich, NJ	
☐ 713-862-1624	RCP/M RBBS Pegasus, Houston, TX	24h
☐ 614-837-3269	RCP/M RBBS, Pickerington, OH	
☐ 415-965-4097	RCP/M RBBS Piconet, Mountain View, CA	
☐ 303-598-3995	RCP/M RBBS, Pinecliffe, CO	24h ★
☐ 716-425-1785	RCP/M RBBS, Rochester, NY	24h ★
☐ 201-932-3879	RCP/M RBBS Rutgers, New Brunswick, NJ	24h
☐ 619-273-4354	RCP/M RBBS, San Diego, CA	24h ★
☐ 408-287-5901	RCP/M RBBS San Jose Oxgate, San Jose, CA	24h
☐ 619-461-0111	RCP/M RBBS SDCS HEC#04, La Mesa, CA	●
☐ 619-236-0742	RCP/M RBBS SDCS, San Diego, CA	24h
☐ 313-559-5326	RCP/M RBBS, Southfield, MI	24h
☐ 604-584-2543	RCP/M RBBS, Surrey, BC, CAN	24h
☐ 813-831-7276	RCP/M RBBS, Tampa, FL	
☐ 313-729-1905	RCP/M RBBS, Westland, MI	
☐ 914-679-8734	RCP/M RBBS, Woodstock, NY	24h ★
☐ 206-458-3086	RCP/M RBBS Yelm, Olympia, WA	
☐ 415-552-9968	RCP/M Rich & Famous, San Francisco, CA	24h
☐ 619-534-1547	RCP/M, San Diego, CA	24h ★
☐ 713-469-8893	RCP/M Satsuma, Houston, TX	● ★
☐ 408-246-5014	RCP/M, Silicon Valley, CA	24h
☐ 805-527-2219	RCP/M, Simi Valley, CA	●
☐ 914-679-6559	RCP/M SJBBS, Bearsville, NY	24h
☐ 607-797-6416	RCP/M SJBBS, Johnson City, NY	●
☐ 1 0-997-1018	RCP/M Software Tools, Sydney, Australia	24h
☐ 408-730-8733	RCP/M, Sunnyvale, CA	●
☐ 617-862-0781	RCP/M Superbrain, Lexington, MA	24h ★
☐ 416-232-0269	RCP/M System One, Mississauga, ON, CAN	24h $ ★
☐ 416-231-1262	RCP/M System Two, Mississauga, ON, CAN	24h $ ★
☐ 713-522-3805	RCP/M Technical, Houston, TX	
☐ 805-492-5472	RCP/M Technical, Thousand Oaks, CA	24h ★
☐ 201-625-1797	RCP/M The C-Line, NJ	●
☐ 604-873-4007	RCP/M Vancouver, BC, CAN	24h
☐ 513-435-5201	RCP/M W. Carrolton, Dayton, OH	24h
☐ 415-941-1990	Realm of the Rogues	
☐ 601-992-1918	Remote Apple Jackson, MS	24h
☐ 404-926-4318	Remote Northstar, Atlanta, GA	24h
☐ 303-444-7231	Remote Northstar, Denver, CO	
☐ 813-381-2394	Remote Northstar, Largo, FL	24h
☐ 301-344-9156	Remote Northstar Nasa, Greenbelt, MD	
☐ 805-964-4115	Remote Northstar, Santa Barbara, CA	
☐ 804-340-5246	Remote Northstar, Virginia Beach, VA	
☐ 401-944-4689	RI Tandy Users Group, Cranston, RI	24h
☐ 401-521-1998	RIAMIS Atari, Providence, RI	24h
☐ 713-497-5433	RIBBS, Houston, TX	
☐ 401-456-8250	RICAMIS, Kingston, RI	24h
☐ 303-279-5657	Robotics-BBS	
☐ 414-462-2225	Rogue Moon	6pm-10am wknds
☐ 616-693-2648	RS-CPM, Clarksville, MI	
☐ 414-476-8010	RSTS	
☐ 416-884-6198	RTC BBS, Richmond Hill, ON, CAN	8pm-9am

S

☐ 618-451-1041	Sattelite/Cable Net	
☐ 512-494-0285	SATUG BBS, San Antonio, TX	
☐ 604-438-2468	Satyricomp, BC, CAN	
☐ 206-763-8879	Seacomm-80, Seattle, WA	24h
☐ 204-785-8742	Selkirk BBS, Selkirk, MB, CAN	24h
☐ 713-777-8608	Shadow World	
☐ 914-359-1517	Sherwood Forest II	
☐ 201-233-5997	Sherwood Forest	
☐ 408-739-5370	Shoalin Temple, Sunnyvale, CA	
☐ 702-826-7277	Signon, Reno, NV	★ pswd = free
☐ 212-442-3874	Sister, Staten Island, NY	24h
☐ 804-285-0041	Skeleton Island	
☐ 618-797-0656	Skull Island V	
☐ 604-584-2731	SMUG, BC, CAN	
☐ 713-453-7931	SOBBS Poor Man's BBS, Houston, TX	24h
☐ 713-522-5516	SOBBS Test Mode, Houston, TX	
☐ 707-576-1478	Software 1st BBS	
☐ 713-468-0198	Software House, Houston, TX	
☐ 603-625-1919	Software Referral Service	
☐ 213-473-2754	Softworx	
☐ 217-875-5579	South Pole	
☐ 312-677-7140	South Pole	
☐ 713-568-6595	Space Voyage, Houston, TX	
☐ 203-834-0026	Spectre-80	
☐ 408-867-4455	Split Infinity, Saratoga, CA	
☐ 707-523-1736	SRCC ABBS, Santa Rosa, CA	
☐ 802-862-7023	ST80-CC Lance Micklus, Inc., Burlington, VT.24h	
☐ 914-782-7605	ST80-PBB Monroe Camera Shop, Monroe, NY	
☐ 703-342-1800	Star City	
☐ 318-237-3350	Star Link	
☐ 602-833-0740	Stellar III, Phoenix, AZ	24h
☐ 913-648-5301	Steve's BBS	24h
☐ 408-338-9511	Stewart II	
☐ 414-762-6411	S.U.E	24h $
☐ 415-452-0350	Sunrise Omega-80, Oakland, CA	
☐ 416-839-3260	Superboard, Pickering, ON, CAN	9pm-8am
☐ 703-765-2161	Switchboard, Alexandria, VA	24h
☐ 415-895-0699	System/80, San Leandro, CA	
☐ 602-861-4090	System-X, Phoenix, AZ	↔

T

☐ 303-690-4566	TBBS, Aurora, CO	
☐ 512-385-1102	TBBS, Austin, TX	24h
☐ 414-281-0545	TBBS Canopus, Milwaukee, WI	24h
☐ 713-442-7644	TBBS Exidy 2000, Houston, TX	24h ★
☐ 713-331-2599	TBBS Freelancin' Alvin, Houston, TX	24h ★
☐ 713-488-2003	TBBS Freelancin', Houston, TX	24h ★
☐ 214-769-3036	TBBS, Hawkins, TX	24h ★
☐ 415-490-8083	TBBS Noah's Ark, Fremont, CA	24h ∞
☐ 305-645-5543	TBBS Pizza-Net, Orlando, FL	24h
☐ 318-635-8660	TBBS, Shreveport, LA	24h
☐ 918-749-0059	TBBS, Tulsa, OK	24h
☐ 212-799-4649	TCBBS Astrocom, New York, NY	24h
☐ 212-362-1040	TCBBS B.A.M.S. New York, NY	24h
☐ 703-836-0384	TCUG BBS, Washington, DC	24h
☐ 414-649-8326	Team (TIBBS)	24h
☐ 301-565-9051	Tech-Link, Forest Glen, MD	24h
☐ 813-839-6746	Tecom-80, Tampa, FL	
☐ 203-746-5763	Telcom 7, New Fairfield, CT	24h
☐ 707-996-2427	Tel-Com	
☐ 414-542-2102	TeleCommunicator's Edge	
☐ 214-960-7654	Teledunjon III	
☐ 404-962-0616	Telemessage-80, Atlanta, GA	
☐ 914-623-4248	Teleport 64	
☐ 305-798-1615	Temple Toa-Rin	
☐ 617-863-0282	TermExec Newsletter, Lexington, MA	
☐ 303-427-7114	Testing Zone	
☐ 817-283-3886	Texas Connection	
☐ 201-994-9620	The Barn, Livingston, NJ	
☐ 414-282-9308	The Connection, Milwaukee, WI	24h
☐ 512-443-3084	The Diner, Austin, TX	
☐ 305-393-7122	The Freezer	
☐ 213-447-0681	The Frigate	
☐ 612-454-6209	The Grapevine	
☐ 414-541-0224	The Milwaukee BBS, Milwaukee, WI	24h
☐ 313-453-9183	The Monitor, Detroit MI	
☐ 304-372-4486	The Morg	
☐ 512-477-2672	The Paradise	
☐ 714-535-7527	The Simarillion, Garden Grove, CA	
☐ 409-765-8866	The Treasure	
☐ 512-441-9429	Thieve's Den	
☐ 416-232-2644	THUG, Mississauga, ON, CAN	7pm-7am
☐ 313-855-6006	Timewarp	
☐ 416-451-7137	TMUG, Brampton, ON, CAN	
☐ 313-453-5146	T-Net Central Processing Unit	24h
☐ 609-896-2436	T-Net Delta Connection	24h
☐ 313-855-6321	T-Net Special Corp	24h
☐ 313-775-1649	T-Net Twilight Phone, Warren, MI	24h
☐ 419-867-9777	Toledo Apple Users BBS, Toledo, OH	24h
☐ 416-782-9534	Toronto PET Users Group BBS (TPUG), Toronto, ON, CAN	24h Ⓟ
☐ 213-375-6137	Torture Chamber, Los Angeles, CA	
☐ 618-234-4243	TPS Network	
☐ 912-439-7440	Trade-80, Albany, GA	24h
☐ 814-898-2952	Trade-80, Erie, PA	24h
☐ 305-525-1192	Trade-80, Ft. Lauderdale, FL	
☐ 402-292-6184	Trade-80, Omaha, NE	
☐ 414-272-0369	Traders Alley, Milwaukee, WI	24h $
☐ 617-443-7428	Trading Post II	
☐ 504-291-4970	Trading Post	
☐ 313-547-7903	Treasure Island	
☐ 805-493-1152	Treasure Vault, Thousand Oaks, CA	
☐ 506-357-5668	TRS-80 BBS, Oromocto, NB, CAN	
☐ 416-839-8274	TRS-80 BBS, Pickering, ON, CAN	
☐ 416-668-1851	TRS-80 BBS, Whitby, ON, CAN	
☐ 416-445-1725	Twilight Comm, North York, ON, CAN	
☐ 213-357-2038	Twilight Zone	

U

☐ 303-796-8708	U called it U name it	
☐ 318-367-8860	USS Enterprise	

V

☐ 414-271-7580	Vanmil, Milwaukee, WI	24h
☐ 714-547-6220	Verga 80, Costa Mesa, CA	
☐ 713-944-6597	VIC-20 Online, Houston, TX	24h
☐ 215-446-7670	Video Ace	
☐ 215-363-0563	Video Fantasies, Langhorne, PA	
☐ 317-742-7725	Viking Communications	
☐ 617-235-5082	Visiboard, Wellesley, MA	
☐ 602-247-6034	Voyager, Phoenix, AZ	

W

☐ 704-373-7966	WAPABBS, Charlotte, NC	24h
☐ 516-293-8659	Ware-House II	
☐ 202-678-9947	Ware-House III	
☐ 618-345-6638	Warlock's Castle	
☐ 703-560-7803	Washington BBS	
☐ 312-623-2226	Waukegan Library, Waukegan, IL	
☐ 703-328-4443	WCCC	
☐ 713-492-8700	Weekender	
☐ 503-649-7814	West Side Network, Portland, OR	
☐ 313-533-0254	Westside Download, Detroit, MI	
☐ 617-326-4812	Westwood BBS	
☐ 414-781-8653	Whizzz...'s Warez (AE)	
☐ 707-257-6502	Wine Country	
☐ 415-845-4812	Winner's Circle	

X

☐ 513-863-7681	XBBS, Hamilton, OH	24h
☐ 713-495-1422	XIO, Houston, TX	●

Y

☐ 213-859-2735	Ye Pawn Shoppe, Los Angeles, CA	

Computer Clubs

User clubs are very nomadic. The listing may show inactive clubs, but the addresses might still be useful for locating others.

Canada

Alberta

Calgary Commodore Users Group
John Hazard
37 Castleridge Dr., N.E.
Calgary, Alberta
Canada T3J 1P4

CCCC (Canadian Commodore Computer Club)
Roger Olanson
c/o Strictly Commodore
47 Coachwood Place
Calgary, Alberta
T3H 1E1
Canada

Bonnyville VIC Cursors
Ed Wittchen
Box 2100
Bonnyville, Alberta
T0A 0L0 403-826-3992
Canada

British Columbia

VIC-TIMS
Greg Goss
2-830 Helena St.
Trail, BC
V1R 3X2 604-368-9970
Canada

Castlegar Commodore Computer Club
Robert Dooley
SS1, S37, C7
Castlegar, BC
V1N 3H7 604-365-3889
Canada

Commodore Computer Club
PO Box 91164
West Vancouver, BC
V7V 3N6 604-738-3311
Canada

Manitoba

W.P.U.G.
Larry Neufeld
9-300 Enniskillen Ave.
Winnipeg, Manitoba
R2V 0H9
Canada

New Brunswick

C-64 Users Group
Don Shea
PO Box 9
Rothesay, NB
E0G 2W0
Canada

Club 64
Cass Howorth
120 Liverpool St.
Fredericton, NB
E3B 4V5 506-454-9730
Canada

Nova Scotia

Nova Scotia Commodore Computer Group
Phil Cummings
PO Box 3426
Halifax South
Halifax, NS
B3J 3J1
Canada

Ontario

Fledging Barrie User Group (BUG)
58 Steel St
Barrie, Ontario
Canada L4M 2E9

PET Educators Group
PO Box 454
Station A
Windsor, Ontario
Canada N9A 6L7

Commodore Users Club of Sudbury
938 Brookfield Ave.
Sudbury, Ontario
Canada P3A 4K4

Toronto PET Users Group, Inc.
Chris Bennett416-782-8900
1912A Avenue Rd., Ste. 1
Toronto, Ontario
M5M 4A1 416-782-9252
Canada

London Commodore Users Club (LCUC)
Dennis Trankner
28 Barrett Cres.
London, Ontario
N6E 1T5 519-681-5059
Canada

Mr. Walter Scholz
568 Mornington St.
Stratford, Ontario
N5A 5G9 519-271-5704
Canada

D. Lerch
Arva Hackers, Medway High School
Arva, Ontario
N0M 1C0
Canada

Cambridge Commodore Users Group
William McLean
c/o Badcock & Wilcox Canaada Ltd.
581 Coronation
Cambridge, Ontario
N1R 5V3
Canada

Cornwall Computer Club
David King
1510 Second St. East
Cornwall, Ontario
K6H 2C3
Canada

Cambridge Commodore Users Group
William McLean
c/o Badcock & Wilcox Canada Ltd.
581 Coronation
Cambridge, Ontario
N1R 5V3
Canada

PET Users Club
Mr. Brown
Valley Heights Secondary School
Box 159
Langton, Ontario
N0E 1G0
Canada

C-64 Users Group
Susan Timar
1122 Wilson Dr.
Sarnia, Ontario
N7S 3J6 519-542-2534
Canada

Brockville Users Group (B.U.G.)
Bill Maxwell
72 Murray St.
Brockville, Ontario
K6V 2X1
Canada

Quebec

COMVIC
PO Box 1688
St. Laurent
Montreal, Quebec
Canada H4L 4Z2

C-64 Users Group Of Montreal - (C.U.G.O.M)
Gary Letovsky
Snowdon PO Box 792
Montreal, Quebec
H3X 3X9
Canada

Saskatchewan

Compu-Dom of Southern Saskatchewan
Joel Champagne
308 Coldwell Rd.
Regina, Saskatchewan
S4R 4L5
Canada

The Regina Commodore Club
K.H. Jones
76 Dolphin Bay
Regina, Saskatchewan
S4S 4Z8 584-2968
Canada

United States

Alaska

Alaska 84 Computer Club
c/o Line 49 Management
PO Box 6043
Anchorage, AK
99502

COMPOOH-T
PO Box 118
Old Harbor, AK
99643 907-286-2213

First City Users Group
James Llanos
PO Box 6692
Ketchikan, AK
99901 907-225-5695

1st City Users Group
James Llanos
PO Box 6692
Ketchikan, AK
99901 907-225-5695

Alabama

Shoals Commodore Users Group (SCUG)
G. Taylor
209 Lakeshore Dr.
Muscle Shoals, AL
35661

William Autry
1734 S. Atmore Ave.
Whistler, AL
36612 205-452-9740

Howard Crider
1920-A Avenue C
Brookly
Mobile, AL
36615 205-661-1973

Wiregrass Micro-Computer Society
Bill Brown
Commodore SIG
109 Key Bennd Rd.
Enterprise, AL
36330 205-347-7564

Commodore Club of Mobile
Tom Wyatt
3868-H Rue Maison
Mobile, AL
36608 205-343-1178

CC & Me
Bill Freeman
PO Box 324
Pinson, AL
35126 205-854-0650

Riverchase Commodore Users Group
Ken Browning
617 Grove St.
Birmingham, AL
35209 205-988-1078

Tiger Byte: E. Alabama CBM 64 Users Group
Jack Parsons
c/o The Computer Store, Inc.
Midway Plaza
Opelika, AL
36801

Huntsville PET Users Club
Hal Carey
9002 Berclair Rd.
Huntsville, AL
35802

The Birmingham Commodore Computer Club
Harry Jones
Birmingham, AL

Arkansas

Booneville 64 Club
Mary Taff
c/o A.R. Hederich Elem. School
401 W. 5th St.
Booneville, AR
72927

Commodore/PET Users Club
Geneva Bowlin
Conway Middle School
Davis St.
Conway, AR
72032

The Siloam Commodore Computer Club
Ken Emanualson
PO Box 88
Siloam Springs, AR
72761 501-524-5624

Arkansas River Valley Commodore Users
Bob Brazeal
401 S. Arlington Dr.
Russellville, AR
72801 501-967-1868

Commodore Computer Club of Ft. Smith, AR
Joe Ragsdale
PO Box 6000
So. Station
Ft. Smith, AR
72906

P.I.C. Club
Bob Reed
c/o Hatfield Public Schools
Box 130
Hatfield, AR
71945 501-389-6164

Arizona

VIC Users Group
Paul Muffuletto
2612 E. Covina
Mesa, AZ
85203

ACUG
Dan Deacon
c/o Home Computer Service
2028 W. Camelback Rd.
Pheonix, AZ
85015 602-249-1186

Catalina Commodore Computer Club
George Pope
2012 Avenida Guillermo
Tucson, AZ
85710 602-296-6766

West Mesa VIC
Kenneth Epstein
2351 S. Standage
Mesa, AZ
85202

Arizona VIC 20-64 Users Club
Donald Kipp
232 W. 9th Place North
Mesa, AZ
85201

Central Arizona PET People
Roy Schaher
842 W. Calle del Norte
Chandler, AZ
85224 602-899-3622

Arizona VIC & 64 Users
Tom Monson
904 W. Marlboro Circle
Chandler, AZ
85224 602-963-6149

Canyon De Chelly - Four Corners Users Group
Larry DiLucchio
c/o Calumet Consulting
Box 1945
Chinle, AZ
86503 602-674-3421

California

The Valley Computer Club
2006 Magnolia Blvd.
Burbank, CA
91506

San Diego Commodore (PET) User Group
Jane Campbell
Box 86531
San Diego, CA
92138 619-277-7214

SIG (Special Interest Group)
Brian R. Klotz
1135 Coronet Ave.
Pasadena, CA
91107

Sixty Fourum
John Damiano
PO Box 16098
Fresno, CA
93755

Pomona Valley Vic Users Group
Mark Joerger
1401 W. 9th, #77
Pomona, CA
91766 714-620-8889

Valley Computer Club
PO Box 310
Denair, CA
95316

Southern California PET Users Group
c/o Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA
90241 213-923-9361

Port Townsend Computer Club
Doug Nash
PO Box 233
Port Townsend, CA
98368

The Exchange
Michael C. Joseph, MD
PO Box 9189
Long Beach, CA
90810 213-595-1771

Walnut Creek PET Users Club
1815 Ygnacio Valley Rd.
Walnut Creek, CA
94596

Jurupa Wizards
Walter J. Scott
8700 Galena St.
Riverside, CA
92509 781-1731

Robyn Graves
8120 Sundance Dr.
Orangevale, CA
95662 916-969-2028

Commodore 64 West Computer Club
Don Campbell
2917 Colorado Ave.
Santa Monica, CA
90404 213-828-9308

PET on the Air
Max J. Babin, Secretary
525 Cresttake Dr.
San Francisco, CA
94132

Diablo Valley Commodore Users Group
PO Box 27155
Concord, CA
94520 415-838-2838

San Fernando Valley Commodore Users Group
Tom Lynch
21208 Nashville
Chatsworth, CA
91311 213-709-4736

Antelope Valley Commodore Users Group
James Haner
POB 4436
Lancaster, CA
93539 805-942-2626

Bay Area Home Computer Asso.
Cliff Downing
1332 Pine St.
Walnut Creek, CA
94598 415-932-5447

San Francisco Commodore Users Group
Roger Tierce
278 - 27th Ave. #103
San Francisco, CA
94121 415-387-0225

Commodore Users Group
Gilbert Vela
4237 Plumeria Ct.
Santa Maria, CA
93455 805-937-4174

Commodore Users Group of Riverside (CUGR)
Ken Brown
PO Box 8748
Riverside, CA
92515 714-689-1452

Marin Commodore Computer Club
620 Del Ganado Rd.
San Rafael, CA

Lincoln Computer Club
John Fung, Advisor
750 E. Yosemite
Manteca, CA
95336

NVCUG
Jim Banks, Jr.
PO Box 1925
Chico, CA
95927 916-343-4611

Sacramento Commodore Users Group
Robyn Graves
8120 Sundance Dr.
Orangevale, CA
95662 916-969-2028

PALS (PETs Around Livermore Society)
J. Johnson
886 South K
Livermore, CA
94550 415-449-1084

SPHINX
Bill MacCracken
267 Arlington Ave.
Kensington, CA
94707 415-527-9286

Commodore Tech. Users Group C-TUG
PO Box 1497
Costa Mesa, CA
92626

Sixty Fourum
Deb Christensen
4413 E. Iowa
Fresno, CA
93702 209-252-0392

C-64/VIC 20 Users Group
Chuck Cypher
Pasadena City College
Cicadian Room
Pasadena, CA

20/64 Users Group
Don Cracraft
PO Box 18473
San Jose, CA
95158

Peninsula Commodore Users Group
Timothy Very
549 Old County Rd.
San Carlos, CA
94070 415-593-7697

VIC-Club: San Francisco (VCSF)
Colin Johnston
1503A Dolores
San Francisco, CA
94110

Humboldt Commodore Group
R. Turner
c/o R. Turner
PO Box 570
Arcata, CA
95521

Commodore 64 West
Charles P. Santos
PO Box 346
Culver City, CA
90232 213-398-0913

20/64
PO Box 18473
San Jose, CA
95158 408-978-0546

PALS (Pets Around Livermore Society)
John Rambo
886 South K
Livermore, CA
94550

Commodore Interest Association
Mark Finley
c/o Computer Data
14660 La Paz Dr.
Victorville, CA
92392

VIC 20 Software Exchange
Vincent Beltz
7660 Western Ave.
Buena Park, CA
90620

Software 64
Mario Abad
353 California Dr.
Burlingame, CA
94010 415-340-7115

Amateurs and Artesians Computing
PO Box 682
Cobb, CA
95426

PUG of Silicon Valley
22355 Rancho Ventura Rd.
Cupertino, CA
95014

VIC 20 Software Exchange Club
Daniel Upton
10530 Sky Circle
Grass Valley, CA
95945

Southern California Edison Commodore Club
Jerry Van Norton
PO Box 800
Rosemead, CA
91770

S.D. East County C-64 User Group
Linda Schwartz
c/o Linda Schwartz
6353 Lake Apopka Place
San Diego, CA
92119 619-698-7814

Manteca VIC 20 Users Organization
Gene Rong
429 N. Main St.
Manteca, CA
95336

Suisun/FF/Vacaville Commodore Users Group
Charles D. Akula
1410 Pelican Way
Suisun City, CA
94585 707-426-2077

Sequoia Computer Users
Dave Demanty
3005 Seeger Avenue
Visalia, CA
93277

South Bay Commodore Users Group
Lloyd Lehrer
401 - 9th St.
Manhatten Beach, CA
90266 213-374-1247

The Diamond Bar R.O.P. Users Group
Don McIntosh
2644 Amelgado
Haciendo Hgts., CA
91745 213-333-2645

CA. Area Commodore Terminal Users Society
Darrell Hall
C.A.C.T.U.S.
PO Box 1277
Alta Loma, CA
91701

VIC TORII-The VIC 20 Users Group
Wesley Clark
PSC #1, Box 23467
APO San Francisco, CA
96230

South Bay Commodore 64 Users Group
PO Box 3193
San Ysidro, CA
95073

C-64 West Orange County Users Group
Philip Putman
PO Box 1457
Huntington Beach, CA
92647 714-842-4484

Santa Rosa Commodore 64 Users Group
Garry Palmer
333 East Robles Ave.
Santa Rosa, CA
95407 707-584-7009

San Luis Obispo Commodore Computer Club
Joan Rinehart
1766 9th St.
Los Osos, CA
93402 805-528-3371

Stockton Commodore Users Group
Andrew Smith
2555 Alexa Way
Stockton, CA
95209 209-478-8419

Computer Using Educators
Leanne Patterson
PO Box 18547
San Jose, CA
95158

LOGIKS Commodore Computer Club
Elmer Johnson
c/o Christ Presbyterian Church
620 Del Ganado Rd.
San Rafael, CA
94903 415-479-0426

Computer Barn Computer Club
S. Mark Vanderbilt
319 Main St.
Suite #2
Salinas, CA
93901 757-0788

Napa Valley Commodore Computer Club
Mick Winter
c/o Liberty Computerware
2680 Jefferson St.
Napa, CA
94558 707-252-6281
night ph. 707-944-2797

The Commodore Connection
Bud Massey
2301 Mission St.
Santa Cruz, CA
95060 408-425-8054

Colorado

VICKIMPET Users Group
Louis Roehrs
4 Waring Lane, Greenwood Village
Littleton, CO
80121

Commodore Users Group
Ray Brooks
Box 377
Aspen, CO
81612 303-925-5604

Vicdore Users Group
Wayne Sundstrom
326 Emery Dr.
Longmont, CO
80501 303-772-2821

Aurora Market Users Group
Roger Oberdier
c/o Computer Market Place
15200 E. 6th Ave.
Aurora, CO
80012 303-367-0901

Colorado Commodore Computer Club
Jack Moss at 986-0577
2187 S. Golden Ct.
Or CONTACT: John Adams at 494-0705.
Denver, CO
80227

Connecticut

John F. Garbarino
Skill Lane Masons Island
Mystic, CT
06355 203-536-9789

New London County Commodore Club
Dr. Walter Doolittle
Doolittle Road
Preston, CT
06360

Fairfield County Commodore Users Group
Linda Retter
PO Box 212
Danbury, CT
06810

Commodore Users Group
Daniel G. Spaneas
Wethersfield High School
411 Wolcott Hill Rd.
Wethersfield, CT
06109

Capitol Region Commodore Computer Club
Prudence Schifley
57 Carter Dr.
Tolland, CT
06084

VIC Users Club
Edward Barszczewski
22 Tunxis Rd.
West Hartford, CT
06107

The Commodore East Users Group
165 B S. Bigelow Rd.
Hampton, CT
06247 203-455-0108

Commodore Users Group of Stratford
Dan Kern-Ekins
PO Box 1213
Stratford, CT
06497 203-377-8373

PEEK & POKE Computer Software Club
Bob J. Pipolo
PO Box 98, 528 Main St.
Cromwell, CT
06416 203-267-2113

CT Computer Society
Harry Hill
180 Bloomfield Ave.
Hartford, CT
06105 203-233-3373

District of Columbia

USO Computer Club
Steven Guenther
USO Outreach Center
207 Beyer Rd., SW
Washington, DC
20332

Delaware

The Diamond State Users Group
Michael Butler
Box 892, RD 2
Felton, DE
19943 302-284-4495

Brandywine Users Group
Rick Jeandell
PO Box 10943
Wilmington, DE
19850 302-362-6162

Newark Commodore Users Group (NCUG)
Bob Black
210 Durso Dr.
Newark, DE
19711 302-737-4686

Florida

South Tampa Commodore 64 Users Group
Ronald S. Clement
736 F Second Dr.
Macdill AFB, FL
33621

Tampa Bay Commodore Computer Club
10208 N. 30th St.
Tampa, FL
33612 813-977-0877

El Shift OH
Mike Schnoke
PO Box 548
Cocoa, FL
32922

Sanibel Commodore Users Group (SCUG)
Phil Belanger
1119 Periwinkle
Box 73
Sanibel, FL
33957 813-472-3471

The Ultimate 64 Experiance
Sandy Cueto
5740 S.W. 56th Terrace
Miami, FL
33143

Tampa Commodore Users Group
PO Box 8713
Tampa, FL
33674 813-237-2100

64 Educators Users Group North
Robert Figueroa
16330 N.E. 2nd Ave.
North Miami Beach, FL
33162 305-944-5548

Ram Rom 84
Nancy Kenneally
1620 Morning Dove Lane
Englewood, FL
33533 813-474-9450

Commodore Users Group
Jim Neill
545 E. Park Ave.
Apt. #2
Tallahassee, FL
32301 904-224-6286

Lakeland VIC 20 Users Group
2450 Shady Acres Dr.
Mulberry, FL
33860

Brandon Users Group
Paul Daugherty
108 Anglewood Dr.
Brandon, FL
33511 813-685-5138

Brandon Commodore Users Group
414 E. Lumsden Rd.
Brandon, FL
33511

64 Educators Users Group South
Dr. Eydie Sloane
FDLRS-South
9220 S.W. 52nd Terrace
Miami, FL
33165 305-274-3501

Miami 20/64
12911 S.W. 49th St.
Miami, FL
33175 305-226-1185

VIC Users Club
Ray Thigpen
4071 Edgewater Dr.
Orlando, FL
32804

PETs and Friends
Richard Plummer
129 NE 44th St.
Miami, FL
33137

South Florida PET Users Group
Dave Young
7170 S.W. 11th St.
West Hollywood, FL
33023 305-987-6982

Commodore Computer Club
David Phillips
PO Box 9726
Jacksonville, FL
32208 904-764-5457

Commodore 64/VIC 20 User Group
Mr. Earl Preston (305)
Martin Marietta Aerospace
PO Box 5837, MP 142
Orlando, FL
32855 352-3252/2266

Gainesville Commodore Users Club
Louis Wallace
3604-20A SW 31st Dr.
Gainesville, FL
32608

Bay Commodore Users Group
Richard Scofield
c/o Gulf Coast Computer Exchange
241 N. Tyndall Pkwy., PO Box 6215
Panama City, FL
32401 904-785-6441

Volusia Ct. Commodore Program Exchange
Rick Stidham
1612 Reynolds Rd.
DeLeon Springs, FL
32028

Suncoast 64s
Curtis Miller
c/o Little Professor Book Center
2395 U.S. 19 North
Palm Harbor, FL
33563 813-785-1036

VIC/64 Heartland Users Group
Tom Keough
1220 Bartow Rd. #23
Lakeland, FL
33801 813-666-2132

Charlotte County Commodore Club (CCCC)
Lee Truax
567 N. Ellicott Circle
Port Charlotte, FL
33952 813-625-1277

Broward Commodore Users Group
Lewis Horn
13 Spinning Wheel Lane
Tamarac, FL
33319 305-726-4390

Richard Prestien
6278 SW 14th St.
Miami, FL
33144

Commodore Computer Club
Chuck Fechko
PO Box 21138
St. Petersburg, FL
33742 813-522-2547

The Class of 64
Joe Statafora
c/o The Computer Corner
5208 - 66th St., North
St. Petersburg, FL
33709 813-541-1185

Jacksonville Area PET Society
401 Monument Rd. #177
Jacksonville, FL
32211

Sun Coast VICs
Mark Weddell
PO Box 1042
Indian Rocks Beach, FL
33535

The Commodore Advantage
Deanna Owens
PO Box 18490
Pensacola, FL
32523 904-456-6554

Clearwater Commodore Club
Gary Gould
1532 Lemon St.
Clearwater, FL
33516 813-442-0770

Commodore Connection
PO Box 6684
West Palm Beach, FL
33405

The Commodore Connection
PO Box 6684
West Palm Beach, FL
33405

Gainesville Commodore Users Group
James E. Birdsell
Santa Fe Community College
Gainesville, FL
32602

Georgia

Atlanta Commodore 64 Users Group
Ron Lisoski
1767 Big Valley Lane
Stone Mountain, GA
30083 404-981-4253

VIC Educators Users Group
Dr. Al Evans
Cherokee County Schools
110 Academy St.
Canton, GA
30114

VIC-tims
Eric Ellison
PO Box 467052
Atlanta, GA
30346 404-922-7088

Atlanta 64 Users Group
Phil J. Autrey
PO Box 5322
Atlanta, GA
30307

Albany Commodore Amateur Computerist
David Via
PO Box 5461
Albany, GA
31706

Commodore Club of Agusta
David Dumas
1011 River Ridge Rd
Apt. #14-A
Agusta, GA
30909

Golden Isles Commodore Users Club
Richard L. Young
Bldg. 68, FLETC
Glynco, GA
31524

Atlanta Computer Society
PO Box 888771
Atlanta, GA
30356

Hawaii

Commodore Users Group of Honolulu
c/o PSH
824 Bannister St.
Honolulu, HI
Meets at Kaliho Library

Commodore Users Group of Honolulu
Jay Calvin 808-944-9380
1626 Wilder #701
Honolulu, HI
96822 808-848-2088

20/64 Hawaii
Wes Goodpaster
PO Box 966
Kailua, HI
96734

Iowa

Commo-Hawk Commodore Users Group
Vern Rotert
PO Box 2724
Cedar Rapids, IA
52406

Quad City Commodore Computer Club
Mike Hoeper
PO Box 3994
Davenport, IA
52808 319-242-1496

Newton Commodore Users Group
David Schmidt
320 W. 9th St., S.
Newton, IA
50208 515-792-0814

Commodore Computer Users Group of Iowa
Laura Miller 515-287-1378
Box 3140
Des Moines, IA
50316 or515-263-0963

Commodore Users Group
114 8th St.
Ames, IA
50010

Siouxland Commodore Club
Gary Johnson
2700 Sheridan St.
Sioux City, IA
51104 712-258-7903

VIC 20 & C-64 User Group
Frederick Volker
421 W. 6th St
Waterloo, IA
50702 319-232-1062

Computer Club
Don Groves
1101 South 2nd Avenue
Marshalltown, IA
50158

Idaho

S.R.H.S. Computer Club
Barney Foster
c/o Salmon River High School
Riggins, ID
83549

GHS Computer Club
Don Kissinger
c/o Grangeville High School
910 S. D St.
Grainesville, ID
83530

Eagle Rock Commodore Users Group
Nancy J. Picker
900 S. Emerson
Idaho Falls, ID
83401

64-B.U.G. (Boise Users Group)
Rick Ohnsman
403 Thatcher St.
Boise, ID
83702 208-384-1423

U.G.L.I.-User Groups of Lower Idaho
Sean Brixey, President
Rt 4
Rupert, ID
83350

Pocatello Commodore Users Group
Richard Harker
1250 E. Benton
Pocatello, ID
83201 208-232-1607

64 BUG (Boise Users Group)
John Rosecrans
PO Box 276
Boise, ID
83701 208-344-6302

Commodore Users Group
Grant Bewick
310 Emerald Dr.
Kellogg, ID
83837 208-784-8751

Illinois

The Commodore 64 Users Group
Gus Pagnotta
Suite 100
4200 Commerce Court
Lisle, IL
60532 312-369-6525

Chicago Commodore 64 Users & Exchange Club
Jim Robinson
PO Box 14233
Chicago, IL
60614

RAP 64/VIC Regional Asso. of Programmers
Bob Hughes
10721 S. Lamon
Oak Lawn, IL
60453

Commodore 64 Users Club
Doyne Horsley
104 Susan Lane
Carterville, IL
62918 618-985-4710

Fox Valley 64 Users Group
Frank Christensen
PO Box 28
No. Aurora, IL
60542 312-898-2779

COMCOE (Commodore Club of Evanston)
Jim Salsbury
2108 Sherman Ave.
Evanston, IL
60201

PAPUG - Peoria Area PET Users Group
Max Taylor
6 Apple Tree Lane
East Peoria, IL
61611 309-673-6635

Rockford Area PET Users Group
1608 Benton St.
Rockford, IL
61107

PET VIC Club (PVC)
Paul Schmidt
40 S. Lincoln
Mundelein, IL
60060

Commodore Users Club
David E. Lawless
1707 East Main St.
Olney, IL
62450

Springfield PET Users Group (SPUG)
Bill Eardley
3116 Concord
Springfield, IL
62704 217-753-8500

Oak Lawn Commodore Users Group
Bob Hughes
The Computer Store
11004 S. Cicero Ave.
Oak Lawn, IL
60453 312-499-1300

The C-64 Users Group, Inc.
David Tamkin
PO Box 46464
Lincolnwood, IL
60646 312-583-4629

VIC 20/64 Users Support Group
David R. Tarvin
114 S. Clark St.
Pana, IL
62557 217-562-4568

Champaign-Urbana Commodore Users Group
Steve Gast
2006 Crescent Dr.
Champaign, IL
61821 217-352-9681

Central Illinois PET User Group
Jim Oldfield
635 Maple
Mt. Zion, IL
62549 217-864-5320

WIPUG
Edward Mills
Rt. 5, Box 75
Quincy, IL
62301 217-656-3671

Commodore SIG Cache
Herb Swanson
Box C-176
323 S. Franklin, #804
Chicago, IL
60606 312-685-0994

ASM/TED User Group
Brant Anderson
200 S. Century
Rantoul, IL
61866 217-893-4577

Fox Valley PET Users Group
Art DeKneef
833 Willow St.
Lake in the Hills, IL
60102 312-658-7321

Illinois Valley Commodore Users Group
Brian Foster
2330 - 12th St.
Peru, IL
61354 815-223-5141

The Kankakee Hackers
William Brouillet
RR #2, Box 228-H
Kankakee, IL
60901 815-937-1083

Mt. Vernon Commodore Users Group (MVCUG)
PO Box 512
Mt. Vernon, IL
62864

McHenry County Commodore Club
John Katkus
227 East Terra Cotta Ave.
Crystal Lake, IL
60014 815-455-3942

Shelly Wernikoff
2731 N. Milwaukee Ave.
Chicago, IL
60647

Indiana

National VIC 20 Program Exchange
Stephen Erwin
102 Hickory Court
Portland, IN
47371 219-726-4202

The National Science Clubs of America
Brian Lepley orJeff Brown
Commodore Users Division
PO Box 10621
Merrillville, IN
46411

East Central Indiana VIC Users
Stephen Erwin
R.R. #2
Portland, IN
47371

Commodore Owners Of Lafayette (COOL!)
Ross Indelicato
20 Patrick Lane
West Lafayette, IN
47906 317-743-3410

VIC/64 Users Group
Richard Clifton
c/o Delco Remy Div. General Motors
2401 Columbus Ave.
Anderson, IN
46014 317-378-3016

Western Indiana Commodore Users Group
Dennis Graham
912 South Brown Ave.
Terre Haute, IN
47803 812-234-5099

Commodore Computer Club
John Patrick, President
3814 Terra Trace
Evansville, IN
47711 812-477-0739

Commodore Users Group
Mark Bender
1020 Michigan Ave.
Logansport, IN
46947 219-722-5205

Fulton County Commodore Users
Jim Tyler
1705-3 Madison
Rochester, IN
46975 219-223-4430

PET/64 Users
Jerry Brinson
10136 E. 96th St.
Indianapolis, IN
46256 317-842-6353

VIC Indy Club
Fred Imhausen
PO Box 11543
Indianapolis, IN
46201 317-357-6906

East Central Indiana VIC User Group
Stephen Erwin
Rural Route # 2
Portland, IN
47371

Seymour Peekers
Dennis Peters
c/o D&L Camera Shop
108 N. Chestnut
Seymour, IN
47274

National VIC-20 Program Exchange
Stephen Erwin, President
102 Hickory Court
Portland, IN
47371 219-726-4202

Northern Indiana Commodore Enthusiasts
Eric T. Bean
927 S. 26th St.
South Bend, IN
46615

Cardinal Sales
Carol Wheeler
6225 Coffman Rd.
Indianapolis, IN
46268 317-298-9650

Commodore 64 Users Group
Dennis Graham
912 South Brown Ave.
Terre Haute, IN
47803 812-234-5099

CHUG (Commodore Hardware Users Group)
Ted Powell
12104 Meadow Lane
Oaklandon, IN
46236

Computer Workshop VIC 20/64 Club
Mary O'Bringer
282 S. 600 W.
Hebron, IN
46341 219-988-4535

Kansas

Commodore Users Group
Walter Lounsbery
6050 S. 183 St. West
Viola, KS
67149

Wichita Area PET Users Group
Mel Zandler
2231 Bullinger
Wichita, KS
67204 316-838-0518

Salt City Commodore Club
Wendell Hinkson
PO Box 2644
Hutchinson, KS
67501

Walnut Valley Commodore User Group
Bob Morris
1003 S. 2nd St.
Arkansas City, KS
67005

Kansas Commodore Computer Club
Paul B. Howard
101 S. Burch
Olathe, KS
66061

Kentucky

C*BUG - Commodore Bardstown User Group
Patrick Kirtley
PO Box 165
Bardstown, KY
40004 502-348-6380

Louisville Users of Commodore KY. (LUCKY)
PO Box 22244
Louisville, KY
40222 502-425-2847

Glasgow Commodore Users Group
Steve England
PO Box 154
Glasgow, KY
42141

The Bowling Green Commodore Users Group
Alex Fitzpatrick
Route 11, Creekside Apt. #6
Bowling Green, KY
42101 502-781-9098

VIC Connection
Jim Kemp
1010 South Elm
Henderson, KY
42420

Louisiana

Franklin Parish Computer Club
James D. Mays, Sr.
#3 Fair Ave.
Winnisboro, LA
71295

Commodore Users Group of Oachita
Beckie Walker
PO Box 175
Swaric, LA
71281 318-343-8044

64-Club News
Tom Parsons
5200 Corporate Blvd.
Baton Rouge, LA
70808 504-925-5870

NOVA
Kenneth McGruder, Sr.
917 Gordon St.
New Orleans, LA
70117 504-948-7643

Commodore 64 Users Group
Richard Hood
PO Box 1422
Baton Rouge, LA
70821

VIC 20 Users Group
Wayne D. Lowery, R.N.
5064 Bowden St.
Marrero, LA
70072 504-341-5305

Ark-La-Tex Commodore 64 Club
Bill Walker
5515 Fairfax
Shreveport, LA
71108 318-636-3611

Massachusetts

Raytheon Commodore Users Group
John Rudy
Raytheon Company
Hartwell Rd. GRA-6
Bedford, MA
01730

Berkshire Home for Little PET Users
Tim Auxier
401 Pomeroy Ave.
Pittsfield, MA
01201

Cape Cod 64 Users Group
Jim Close
358 Forrest Rd.
S. Yarmouth, MA
02664 1-800-225-7136

VIC Interface Club
Bernie Robichaud
48 Van Cliff Ave.
Brockton, MA
02401

The Boston Computer Society
Mary E. McCann
Three Center Plaza
Boston, MA
02108 617-367-8080

EM 20/64 Users Group
John Chaplain
36 Buckman St.
Woburn, MA
01801

Eastern Massachusetts VIC Users Group
Frank Ordway
7 Flagg Rd.
Marlboro, MA
02173

Pioneer Valley VIC Club
Marvin Yale
34 Bates Ave.
Westfield, MA
01085 413-562-1027

Berkshire PET Lovers CBM Users Group
Taconic High
Pittsfield, MA
01201

Commodore Users Group
c/o Best Business Equipment
269 Lincoln St.
Worcester, MA
01605

The Cursor Club
John
442 Mulpuf Rd.
Lunenburg, MA
01462 617-582-4056

Masspet Commodore Users Group
Harry Flaxman
PO Box 283
Taunton, MA
02780

Pioneer Valley VIC/64 Club
Marvin Yale
34 Bates St.
Westfield, MA
01085 413-562-1027

Commodore 64 Users Group of The Berkshires
Ed Rucinski
184 Highland Ave.
Pittsfield, MA
01201

VIC Users Group
c/o Ilene Hoffman-Sholar
Needham, MA
02192

CUG of MA.
Paul & Jenny
1132 N. Ridge Rd.
Westfield, MA
01085 413-568-2228

Commodore Users Club
Mike Lennon
Stoughton High School
Stoughton, MA
02072

Maryland

VIC & 64 Users Group
Tom DeReggi
The Boyds Connection
21000 Clarksburg Rd.
Boyds, MD
20841 301-428-3174

Harford County Commodore Users Group
Kim Loyd
PO Box 209
Fallston, MD
21047 301-879-3583

Blue TUSK
Jim Hauff
700 East Joppa Rd.
Baltimore, MD
21204

Long Lines Computer Club
Gene Noff
323 N. Charles St., Rm. 201
Baltimore, MD
21201

Commodore 64 Users Group
Jorge Montalvan
11209 Tack House Court
Potomac, MD
20854 301-983-8199

The Compucats' Commodore Computer Club
Betty Jane Schueler
680 W. Bel Air Ave.
Aberdeen, MD
21001 301-272-0472

House of Commodore
Ernest J. Fischer
8835 Satyr Hill Rd.
Baltimore, MD
21234

Jumpers 2064s (Glen Burnie)
Walt Marhefka
7837 B&A Blvd.
Glen Burnie, MD
21061 301-768-1892

Bay-Cug - Baltimore Area Commodore Users
Michael M. Broumberg
4605 Vogt Ave.
Baltimore, MD
21206 301-325-2156

Rockville VIC/64 Users Group
Tom Pounds
PO Box 8805
Rockville, MD
20856 301-231-7823

Assoc. of Personal Computer Users
5014 Rodman Rd.
Bethesda, MD
20016

Westinghouse BWI Commodore User Group
Attn: L. Barron Mail Stop 5320
PO Box 1693
Baltimore, MD
21203

HUG (Hagerstown Users Group)
Joseph Rutkowski
23 Conventry Lane
Hagerstown, MD
21740 301-797-9728

Gaithersburg C-64 Users Group
Russel Jarosinski
12937 Pickering Dr.
Germantown, MD
20874 301-428-3328

Commodore Users Group of Annapolis
The Software Co.
PO Box 9726
Arnold, MD
21012 301-974-4548

Edison Commodore Users Group
Bill Foley
4314 Oxford Dr.
Suitland, MD
20746 301-423-7155

VIClique (Linthicum Heights)
Pat Foley
105A Conduit St.
Annapolis, MD
21401 301-263-8568

The Montgomery Ct. Commodore Computer Soc.
Meryle Pounds
PO Box 6444
Silver Springs, MD
20906 301-946-1564

Southern MD Commodore Users Group
Tom Helmke
6800 Killarney St.
Clinton, MD
20735 301-868-6536

Maine

So. ME. 64
Ed Moore
10 Walker St.
Portland, ME
04102 207-761-1626

Compumania
Richard L. Nadeau
81 North St.
Saco, ME
04072 207-282-7418

Your Commodore Users Group
Mike Procise
Box 611
Westbrook, ME
04092 207-854-4579

Northwoods Commodore Users Group
Diane Porter
740 Main St.
Van Buren, ME
04785

COM-VICS (Commodore/VIC Users Group)
Paul Lodge
RFD #1, Box 2086
Hebron, ME
04238 207-966-3641

Michigan

C.A.T.O.
Dean Tidwell
17606 Valade
Riverview, MI
48192

Commodore Computer Club
John Walley
4106 Eastman Rd.
Midland, MI
48640 517-835-5130

VIC Users Club
John Gannon
University of Michigan
School of Public Health
Ann Arbor, MI
48109

Commodore Users Group
Albert Meinke, III, M.D.
c/o Eaton Rapids Medical Clinic
101 Spicerville Hwy.
Eaton Rapids, MI
48827

South East Michigan PET Users Group
Norm Eisenberg
Box 214
Farmington, MI
48024

South Computer Club
Ronald Ruppert
South Jr. High School
45201 Owen
Belleville, MI
48111

Commodore Users Group
c/o Family Computer
3947 W. 12 Mile Rd.
Berkley, MI
48072

DEBUG
Herbert Edward
PO Box 196
Berrien Springs, MI
49103 616-471-1882

DAB Computer Club
Dennis Burlingham
PO Box 542
Watervliet, MI
49098 616-463-5457

SMCUG
Dean Otto
1002 Pfau St.
Mankato, MI
56001 507-625-6942

Jackson Commodore Computer Club
Alfred Bruey
201 S. Grinnell St.
Jackson, MI
49203

David Liem
14361 Warwick St.
Detroit, MI
48223

Commodore User Club
Robert Steinbrecher
32303 Columbus Dr.
Warren, MI
48093

Michigan's Commodore 64 Users Group (MCUG)
William G. Osipoff
PO Box 539
E. Detroit, MI
48021 313-773-6302

Mid-Michigan Commodore Club
Virgil Graham
Clare, MI

COMP
M. Gauthier
486 Michigan Ave.
Marysville, MI
48040 313-364-6804

VIC, 64, PET Users Group (West Oakland)
Bert Searing
8439 Arlis Rd.
Union Lake, MI
48085 363-8539

Steve Lepsetz 353-1130 or
20050 Winchechester
Southfield, MI
48076 313-354-7224

Slipped Disk, Inc.
31044 John R
Madison Heights, MI
48071 313-583-9803

Commodore Computer Club of Toledo
Gerald Carter
734 Donna Dr.
Temperance, MI
48182

West Michigan Commodores
Gene Traas
c/o R. Taber
1952 Cleveland Ave., S.W.
Wyoming, MI
49509 616-458-9724

Ann Arbor Commodore Users Group
Art Shaw
Ann Arbor, MI
48103 313-994-4751

SEM 64
Gary Groelier
25015 Five Mile #3
Redford, MI
48239 313-537-4163

Michigan's Commodore 64 Users Group
PO Box 539
East Detroit, MI
48021 313-772-6302

VIC for Business
Mike Marotta
6027 Orchard Ct.
Lansing, MI
48910

Minnesota

Lake Superior Commodore
Peter Roufs
1936 Lawn St.
Duluth, MN
55812 218-728-3224

Twin Cities Commodore Computer Club
Rollie Schmidt
6623 Ives Lane
Maple Grove, MN
55369 612-424-2425

Heartland Area Computer Cooperative
Robert Walz
...a Commodore Computer Club
Route 4, Box 204
Little Falls, MN
56345 612-632-5511

MUPET (Minnisota Users of PET)
Jon T. Minerich
PO Box 179
Annandale, MN
55302

Brainerd Area Commodore Users Group
Norm Saavedra
1219 S.E. 11th St.
Brainerd, MN
56401 218-829-0805

Missouri

MOARK Commodore Users Group
Marshall Turner
PO Box 504
Golden, MO
65658 417-271-3293

The Commodore Users Group of St. Louis
Dan Weidman
Box 6653
St. Louis, MO
63125 314-968-4409

St. Louis Computer Group
Mike Lapusan
5600 Clayton Rd.
St. Louis, MO
63110

Mid-Missouri Commodore Club
Jim Whitacre
780 East Park Lane
Columbia, MO
65201 314-474-2868

KCPUG
Rick West
5214 Blue Ridge Blvd.
Kansas City, MO
64133 816-356-2382

Commodore P.A.C.
Patricia Lucido
Horace Mann Room 202
Maryville, MO
64468 816-582-4498

VIC INFONET
Jory Sherman
PO Box 1069
Branson, MO
65616 417-334-6099

Worth County PET Users Group
David Hardy
Grant City, MO

Joplin Commodore Computers Users Group
R.D. Connely
422 S. Florida Ave.
Joplin, MO
64801

Clearwater Club
Carolyn Polk
Clearwater School
Star Route
Piedmont, MO
63957

Mississippi

Commodore Biloxi Users Group
John Lassen
c/o Universal Computer Services
3002 Hwy. 90 East
Ocean Springs, MS
39564 601-875-1173

Commodore Biloxi User Group (ComBUG)
John Lassen
Universal Computer Services
3002 Hwy. 90 East
Ocean Springs, MS
39564 601-875-1173

Commodore Computer Club
Andrew Holder
Southern Station Box 10076
Hattiesburg, MS
38401 601-268-7585

Montana

Commodore Users Club
Mike McCarthy
1109 West Broadway
Butte, MT
59701

Powder River Computer Club
Jim Sampson
Powder River County High School
Broadus, MT
59317

North Carolina

VIC Users Club
David C. Fonenberry
Route 3, Box 351
Lincolnton, NC
28092

VIC Users Club
Tim Gromlovits
Rt. 11, Box 686
Hickory, NC
28601

Raleigh VIC 20/64 Users Group
Larry Diener
410-D Delta Court
Cary, NC
27511 919-469-3862

Microcomputer Users Club
Joel D. Brown
Box 17142 Bethabara Sta.
Winston-Salem, NC
27116

Down East Commodore Users Groups
Bruce Theden
302 Belltown Rd.
Havelock, NC
28532 919-447-4536

Down East Commodores
Bruce Thedin
302 Belltown Rd.
Havelock, NC
28532 919-447-4536

Cleveland County Computer Club
Todd Patterson
PO Box 489
Grover, NC
28073 704-937-9124

Amateur Radio PET Users Group
Hank Roth
PO Box 30694
Raleigh, NC
27622

Tryon Commodore 64 Club
Robin Michael
PO Box 1016
Tryon, NC
28782 704-859-6340

North Dakota

CCCC (Capitol City Computer Club)
Rolf Arnold
c/o Veterans Memorial Public Library
520 Avenue A East
Bismarck, ND
58501

The Computer Club
Ed Reitan
Lock Drawer 1497
North Dakota State Penitentiary
Bismarck, ND
58502

Nebraska

Marilyn Sallee
1629 Boise
Alliance, NE
69301

Platte Valley Commodore User Group (PVCUG)
Jim Parks
1720 - O - St.
Gering, NE
69341 308-436-3211

National VIC 20 Users Group
George F. Kaywood
PO Box 34575
Omaha, NE
68134

Greater Omaha Commodore 64 Users Group
Bob Quisenberry
2932 Leawood Dr.
Omaha, NE
68123 402-292-2753

New Hampshire

C-64 U.S.E.R.S. User Software Exchange Pro
PO Box 4022
Rochester, NH
03867

TBH VIC-NICs
PO Box 981
Salem, NH
03079

Northern New England Computer Society
PO Box 69
Berlin, NH
03570

New Jersey

The Bell Communication Research
Walter Hobbie
Commodore Users Group
Rm. 17-32 2883, 95 N. Maple Ave.
Basking Ridge, NJ
07920 201-221-4427

Parsippany Computer Group
Bob Searing
51 Ferncliff Rd.
Morris Plains, NJ
07950 201-267-5231

Ewing Commodore Users Group
John C. Jones
11 Van Saun Dr.
Trenton, NJ
08628 609-882-4826

Somerset Users Club
Robert Holzer
49 Marcy St.
Somerset, NJ
08873

Rancocas Valley Users Group
M. Eisenbacher
PO Box 234
Mt. Laurel, NJ
08054 609-267-1912

Cape-Atlantic Commodore Users Group
B.J. Chadwick
1515 Shore Rd.
Lincoln, NJ
08221 398-4044

VIC 20 User Group
G. M. Amin
67 Distler Ave.
W. Caldwell, NJ
07006 201-284-2281

Rancocas Valley Commodore Users Group
Mario Eisenbacher
PO Box 234
Mt. Laurel, NJ
08054 609-267-1912

Educators Advisory
John Hanfield
PO Box 186
Medford, NJ
08055 609-953-1200

VIC-TIMES
Thomas R. Molnar
46 Wayne St.
Edison, NJ
08817

Commodore Friendly User Group
Rich Pinto/Colin Campbell
49 Hershey Rd.
Wayne, NJ
07470 201-696-8043

South Jersey Commodore Users Group
Mark Orthner
c/o Mark Orthner
46B Monroe Path
Maple Shade, NJ
08052 609-667-9758

INFO 64
Dave Garaffa
16 W. Ridgewood Ave.
Ridgewood, NJ
07450 201-447-4422

VIC Software Development Club
H. P. Rosenberg
77 Fomalhaut Ave.
Sewell, NJ
08080

Monmouth Commodore/PET Users Club
Stan Gawel
25 Fox Wood Run
Middleton, NJ
07748 201-671-4059

ACGNJ PET/VIC/CBM User Group
J. M. Pylka
30 Riverview Terr.
Belle Mead, NJ
08502 201-359-3862

Morris Area Commodore Users Group (MACUG)
Tom Limoncelli
61 Early St.
Morristown, NJ
07960 201-267-5088

Bordentown Area Commodore Users Group
Joe Griner
10 Spring St.
Bordentown, NJ
08505 609-298-6275

Jersey Shore Commodore Users Group
201-542-2113 or 223-1387
(Covering Ocean & Monmouth Counties)

New Mexico

Southern New Mexico Commodore Users Group
Scott Gardenhire
2265 N. Dona Ana Rd.
Las Cruces, NM
88005 505-523-5336

Commodore Users Group
Danny Byrne
6212 Karlson, NE
Albuquerque, NM
87113 505-821-5812

Nevada

Las Vegas PET Users Group
Gerald Hasty
Suite 5-315
5130 E. Charleston Blvd.
Las Vegas, NV
89122

C-Run
Franklin Miller
PO Box 70473
Reno, NV
89570

Compu Club 64
Cindy Springfield
4220 S. Maryland Parkway
Bldg. B - Suite 403
Los Vegas, NV
89109 702-369-7354

Southern Nevada Commodore Group
Joseph Windolph
905 Biljac St.
Las Vegas, NV
89128 363-2519

New York

Norny Chug
Andrew VanDuyne
PO Box 226
Norwood, NY
13668 353-4591

PET User Club of Westchester
Ben Meyer
PO Box 1280
White Plains, NY
10602

Queens N.Y. Users Group
Sam Soltan, Bruce Behrend
67-42 Harrow St
Forest Hills, NY

Naples Commodore Users Group
Donald Schmidt
PO Box 11, U.S.N.S.A.
FPO, New York, NY
09521

Commodore 64 Berlin Users Group
Charles D. Blagburn
Co. B USAFS Berlin
Box 9723
APO New York, NY
09742

VIC Users Group
Robert Wurtzel
c/o Stoney Brook Learning Center
1424 Stoney Brook Rd.
Stoney Brook, NY
11790 516-751-1719

LIVE (Long Island VIC Enthusiasts)
Arnold Friedman
17 Picadilly Rd.
Great Neck, NY
11023

Mohawk Valley Commodore Users Group
William Nowak
PO Box 343
Tribes Hill, NY
12177 518-829-7576

Manhatten 64
Larry Thompson
c/o Steve Lazarowitz
1440 Freeport Loop
Brooklyn, NY
11239 212-647-4266

Capitol Dist. 64/VIC 20 Users Group
Bill Pizer
363 Hamilton St.
Albany, NY
12210 518-436-1190

SCUG (Schenectady Commodore Users Group)
Timothy Davis
c/o The Video Connection
Canal Square
Schenectady, NY
12305

Adirondack Commodore 64 Users Group
Paul Klompas
205 Woodlawn Ave.
Saratoga Springs, NY

VIC 20/64 Users Group
Lawrence Schulman
NYU
Waverly Place
New York, NY
10003 212-358-5155

The Upstate Commodore Users Group
Chris Johnson
PO Box 5242
Arnot Mall
Horseheads, NY
14844

Finger Lakes Commodore Users Group
c/o Rose City Computer Associates
229 West Union St.
Newark, NY
14513 315-331-1185

West Chester County VIC Users Group
Joe Brown
PO Box 146
Pelham, NY
10552

New York Commodore Users Group
Ben Tunkelang
380 Riverside Dr., 7Q
New York, NY
10025 212-566-6250

long Island PET Society
Ralph Bressler
Harborfields HS
Taylor Ave.
Greenlawn, NY
11740

Gary Lee Crowell
505-84-6667 E-3S 5th Gen. Hosp.
APO New York, NY
09154

Commodore 64 Users Group
Sam Soltan
67-42 Harrow St.
Forest Hills, NY

New York 64 Users Group
Bruce Cohen
222 Thompson St.
New York, NY
10012 212-673-7241

Commodore Masters
Stephen Farkouh
25 Croton Ave.
Staten Island, NY
10301

The Commodore Users Group Rochester
Tom Werenski
78 Hardison Rd.
Rochester, NY
14617 716-544-5251

VIC 20 User Club
Gary Overman
339 Park Ave.
Babylon, NY
11702 516-669-9126

The New York City VIC/64 Users Group-NYCUG
Joycelyn Woods
436 East 69th St.
New York, NY
10021 212-787-2854

Utica Commodore Users Group
Phil Rothstein
1801 Storrs Ave.
Utica, NY
13501 315-733-2244

SPUG
Paul Skipski
4782 Boston Post Rd.
Pelham, NY
10803

Hudson Valley Commodore Club
PO Box 2190
Kingston, NY
12401

Commodoreee 64 Users Group
John R. Boronkay
S.U.N.Y. at Oswego
Dept. of Industrial Arts
Oswego, NY
13126

VIC Users Club
Michael Frantz
76 Radford St.
Staten Island, NY
10314

Commodore Computer Users Group Heidelberg
Robert H. Jacquot
PO Box, Gen. Del.
APO New York, NY
09102

Commodore SIG Computer Club Of Rockland
Peter Bellin
PO Box 233
Tallman, NY
10982 914-357-8941

VIC Information Exchange Club
Tom Schiegel
336 W. 23 St.
Deer Park, NY
11729 SASE & ph. pl.

VIC 20 User Club
Jean F. Coppola
151-28 22nd Ave.
Whitestone, NY
11357

Rockland County Commodore Users Group
Ross Garber
PO Box 573
Nanuet, NY
10965

Folklife Terminal Club
John Krebs
PO Box 2222-AS
Mt. Vernon, NY
10551

Intercalc (spreadsheet users group)
Bob Korngold
PO Box 254
Scarsdale, NY
10583

LIVICS (Long Island VIC Society)
Lawrence Stefani
20 Spyglass Lane
East Setauketm, NY
11733 516-751-7844

VIC 20 User Group
David Upham, Sr.
Paper Service Division
Kodak Park
Rochester, NY
14617

Bayside VIC Users
Marc Gerstein
23-20 Bell Blvd
Bayside, NY
11360

L&M Computer Club VIC 20 & 64
Dick Mickelson
4 Clinton St.
Tully, NY
13159 315-696-8904

Commodore Computer Club
Neil Threulsen
Publications Dept.,Grumman Aerospace
1111 Stewart Ave.
Bethpage, NY
11714 516-575-9558

VIC 20/64 Users Group
Pete Lobol
31 Maple Dr.
Lindenhurst, NY
11757 516-957-1512

Computer Club of Rockland
Ann Ney
PO Box 233
Tallman, NY
10982 357-7937

Hello, Central!
Jared Sherman
76-12 35th Ave.
Jackson Heights, NY
11372

Commodore Sig Computer Club of Rockland
Peter Bellin
PO Box 233
Tallman, NY
10982 914-357-8941

Poughkeepsie VIC User Group
Joe Steinman
2 Brooklands Farm Rd.
Poughkeepsie, NY
12601 914-462-4518

VIC User Group
Dr. Levitt
1250 Ocean Ave.
Brooklyn, NY
11230 212-859-3030

Ohio

Akron Area C-64 Users Group
Paul Hardy
2453 Second St.
Cuyahoga Falls, OH
44221 216-923-4396

C.P.U. Connection
Danni Hudak
PO Box 42032
Brook Park, OH
44142

S.W.O.C.U.G.(SW. Ohio Commodore Users Gp.)
Joe Beresford
8401 Wicklow Ave.
Cincinnati, OH
45236

Central Ohio PET Users Group
Phillip H. Lynch
107 S. Westmoor Ave.
Columbus, OH
43204 614-274-0304

Medina Commodore Users Group
Jill Carpenter
PO Box 182
Medina, OH
44258 216-722-2611

Marion Ohio Commodore Users Group (MOCUG)
Van Munro
775 Wolfinger Rd.
Marion, OH
43302 614-726-2630

Chillicothe Commodore Users Group
William A. Chaney
PO Box 211
Chillicothe, OH
45601

Paul M. Warner
11433 Pearl Rd.
Strongsville, OH
44136

Amateur Computer Society of Central OH
Jim Crowley
PO Box 28606
Columbus, OH
43228

Commodore Local Users Exchange (C.L.U.E.)
Pat Murphy
3040 Highcliff Ct.
Columbus, OH
43229

Southwestern Ohio Commodore Users Group
PO Box 399117
Cincinnati, OH
45239

Licking County 64 Users Group
323 Schuler St.
Newark, OH
43055 614-345-1327

Commodore Users Group
Carl Skala
18813 Harlan Dr.
Maple Heights, OH
44137 216-581-3099

Dayton Area Commodore Users Group
Charles Tobin
679 Murray Hill Dr.
Xenia, OH
45385 513-372-4077

Commodore Users of Blue Chip (Cincinnati)
Ted Stalets
816 Beecher St.
Cincinnati, OH
45206 513-961-6582

Oklahoma

Commodore Users
Monte Maker, President
Box 268
Oklahoma City, OK
73101

Commodore Users Group
Steve Ford
Muskogee Computer Society
202 S. 12th St.
Muskogee, OK
74401

Commodore Users of Norman
Matt Hager
209 Brookwood
Noble, OK
73068

Southwest Oklahoma Computer Club
c/o Commodore Chapter
PO Box 6646
Lawton, OK
73504

Commodore Oklahoma Users Club
Stanley B. Dow
4000 NW 14th St
Oklahoma City, OK
73107 405-943-1370

Commodore Hobby Users Group (CHUG)
Annette Hinshaw
Box 15238
Tulsa, OK
74158 918-834-5658

Greater Oklahoma Commodore Club
Randy Hill
1401 N. Rockwell
Oklahoma City, OK
73127 405-789-3229

Oregon

United States Commodore Users Group
Richard Tsukiji
PO Box 2310
Roseburg, OR
97470 503-672-7591

NW PET Users Group
John F. Jones
2134 N.E. 45th Ave.
Portland, OR
97213

US Commodore Users Group
Richard Tsukiji
1385 Cleveland Loop Dr.
Roseburg, OR
97470

Southern Oregon VIC/64 Users Group
James Powell
3600 Madrona Lane
Medford, OR
97501 503-779-7631

Jefferson State Computer Users Group-JUG
John Newman
2355 Camp Baker Rd.
Medford, OR
97501

Pennsylvania

G.R.C. User Club
Bill Bolt
300 Whitten Hollow Rd.
New Kensington, PA
15068

Bellwood - Altoona Users Group
D.N. Dantof
1433 - 13th Ave.
Altoona, PA
16603 814-942-9565

Commodore Users Group
Jim Mathers
3021 Ben Venue Dr.
Greensburg, PA
15601 412-836-2224

Commodore Users Group
Matt Matulaitis
781 Dick Ave.
Warminster, PA
18974

VIC 20 Programers, Inc.
Robert Gougher
c/o Watson Woods
115 Old Spring Rd.
Coatesville, PA
19320

Clifton Heights Users Group
PO Box 235
Clifton Heights, PA
19018

VIC Software Development Club
Tracy Lee Thomas
440 W. Sedgwick
Apt. A-1
Philadelphia, PA
19119 215-844-4328

G/C Computer Owners Group
Jo Lambert 215-775-2600
c/o Gilbert Associates, Inc.
PO Box 1498
Reading, PA
19607 Extention 6472

Gene Planchak
4820 Anne Lane
Sharpsville, PA
15150 412-962-9682

The Commodore Users Club of S.E Pittsburgh
Charles Groves
c/o Groves Appliance & TV
2407 Pennsylvania Ave.
West Mifflin, PA
15122

Main Line Commodore Users Group (MLCUG)
Emil Volcheck
1046 General Allen Lane
West Chester, PA
19380 215-388-1581

Oxford Circle 64 User Group
Roger Nazeley215-535-9021
Trinity Church
6900 Rising Sun Ave.
Philadelphia, PA
19111 215-743-8999

Bits & Bytes
Dave Boodey
1015 Dale Rd.
Secane, PA
19018 215-544-5875

CACC (Capitol Area Commodore Club)
Geoffrey Hebert
PO Box 333
Lemoyne, PA
17043 717-732-5255

Penn Conference Computer Club
Dan R. Knepp
c/o Penn Conference of SDA
720 Museum Rd.
Reading, PA
19611

PET User Group
Gene Beals
PO Box 371
Montgomeryville, PA
18936

A-K 64 Users Group
Alton E. Glubish
1762 Fairmont St.
New Kensington, PA
15068 412-335-9070

PACS Commodore Users Group
Stephen Longo
LaSalle College
20th & Olney Ave.
Philadelphia, PA
19141 215-951-1258

Lincoln Technical Inst.
Alan Karpe
5151 Tilghman
Allentown, PA

PPG (Pittsburgh PET Group)
Joel A. Casar, DMD
2015 Garrick Dr.
Pittsburgh, PA
15235 412-371-2882

Westmoreland Commodore Users Club
Jim Mathers
c/o DJ & Son Electronics
Colonial Plaza
Latrobe, PA
15650

Boeing Employees Personal Compute Club
Jim McLaughlin
The Boeing Vertol Co.
PO Box 16858
Philadelphia, PA
19142 215-522-2257

Worldwide Commodore Users Group
David Walter
PO Box 337
Blue Bell, PA
19422

Upper Buxmont C-64 Users
Don Roques
655 Bergey Rd.
Telford, PA
18969 215-723-7039

CACCC-Centre Area Commodore Computer Club
Bill Hillner
214 Computer Building
University Park, PA
16802 814-237-5912

Scranton Commodore Users Group
PO Box 211
Clarks Summit, PA
18411

NADC Commodore Users Club
Norman McCrary
248 Oakdale Ave.
Horsham, PA
19044

MARGA
Mindy Skelton
PO Box 76
Mount Holly Springs PA
17065 717-486-3274

COMPSTARS
Mike Norm
130 Blue Teel Circle
Audubon, PA
19403

Puerto Rico

CUG of Puerto Rico
Ken Burch
RFD #1, Box 13
San Juan, PR
00914

VIC 20 User Group
Robert Morales, Jr.
655 Hernandez St.
Miramar, PR
00907

Rhode Island

Newport VIC/64 Users
Dr. Matt McConeghy
10 Maitland Ct
Newport, RI
02840 401-849-2684

Irving B. Silverman, CPA
Michelle Chavani
160 Taunton Ave.
E. Providence, RI
02914

Commodore Users Group
Victor Moffett
c/o Data-Co.
978 Tiogue Ave.
Coventry, RI
02816 401-828-7385

The VIC 20 Users Club
Tom Davey
Warwick, RI
02886

South Carolina

Spartanburg Commodore Users Group
James Pasley
803 Lucerne Dr.
Spartanburg, SC
29302 803-582-5897

The Charleston Computer Society
Jack Furr
PO Box 5264
N. Charleston, SC
29406 803-747-0310

Lords of BASIC
Robert L. Whisonant
PO Box 459
Ladson, SC
29456

Beaufort Technical College
Dean of Instruction
100 S. Ribaut Rd.
Beaufort, SC
29902

Commodore Computer Club of Columbia
Chuck Howard-Sect./Tres.
PO Box 2775
Cayce
West Columbia, SC
29171

The Executive Touch C-64 & VIC 20 Users
Patricia Watkins
208 Hwy 15
Myrtle Beach, SC
29577 448-8428

Commodore Users Society of Greenville(CUS)
Bo Jeanes
Horizon Records-Home Computers
347 S. Pleasantburg Dr.
Greenville, SC
29607 803-235-7922

South Dakota

VIC/64 Users Club
Larry Lundeen
608 West 5th
Pierre, SD
57501 605-224-4863

PET User Group
Jim Dallas
515 South Duff
Mitchell, SD
57301 605-996-8277

Tennesee

Memphis Commodore Users Club
Harry Ewart
2476 Redvers Ave.
Memphis, TN
38127 901-358-5823

ET 64 Users Group
Walt Turner
PO Box 495
Knoxville, TN
37901 615-966-8478

Jackson Commodore Users Group
Rick Crone
31 Carriage House Dr.
Jackson, TN
38305 901-668-8958

River City Computer
Hobbyists
Memphis, TN

Memphis Commodore Users Group
Harry Ewart
2476 Ridvers Ave.
Memphis, TN
38127 901-358-5823

Nashville Commodore Users Group
Dave Rushing
PO Box 121282
Nashville, TN
37212 615-331-5408

Metro-Knoxville Commodore Users Club
Ed Pritchard
7405 Oxmoor Rd., Rt.# 20
Knoxville, TN
37931 615-938-3773

Commodore User Club
Metro Computer Center
1800 Dayton Blvd.
Chattanooga, TN
37405

Texas

PET Users
2001 Bryan Tower
Suite 3800
Dallas, TX
75201

CHUG (Commodore Houston Users Group)
John Walker
8738 Wildforest
Houston, TX
77088 713-999-3550

Interface Computer Club
M.E. Garza, President
814 North Sabinas
San Antonio, TX
78207

Mid-Cities Commodore Club
Bruce Nelson
413 Chisolm Trail
Hurst, TX
76053

Corpus Christi Commodores
Bob McKelvy
PO Box 6541
Corpus Christi, TX
78411 512-852-7665

PET User Group
John Bowen
Texas A & M
Microcomputer Club
Texas A & M, TX

64 Users Group
Stan Grodin
2421 Midnight Circle
Plano, TX
75075

The Great Northwest CBM 64 Users Group
Randy
6302 War Hawk Dr.
San Antonio, TX
78238 647-3881

VIC Users Group
3817 64th St.
Lubbock, TX
79413

Larry Williams
PO Box 652
San Antonio, TX
78293

Fantasy Commodore Club
Ed Howdershelt
1913 Dalworth St.
Grand Prairie, TX
75050

ICUG (Irving Commodore Users Group)
Robert Hayes
3237 Northgate #1289
Irving, TX
75062 214-252-7017

Commodore Users Group
Danny Miller
624 Bellview St.
Sulphur Springs, TX
75482

VIC 20 Users Group
Jeff Southerland
6416 Brookhaven Trail
Ft. Worth, TX
76133 817-346-1407

Compugild
Johnathan Witt
2211 South Lipscomb
Amarillo, TX
79109

Mid-Cities Commodore Club
Garry Wordelman
413 Chisolm Trail
Hurst, TX
76053

SCOPE
PO Box 3095
Richardson, TX
75083

Gulf Coast Commodore Users Group
Lawrence Hernandez
PO Box 128
Corpus Christi, TX
78403 512-887-4577

James Meeker
1110 Texas Ave.
Mart, TX
76664 817-876-2710

The Woodlands Commodore Users Group
Andrew Gardner
3 Splitrock Rd.
The Woodlands, TX
77380 713-292-8987

Savid Computer Club
Davi Jordan, Chairman
312 West Alabama
Suite 2
Houston, TX
77006

Commodore Users Group (Austin)
Dr. Jerry D. Frazee
PO Box 49138
Austin, TX
78765

64 Users Group
S. G. Grodin
2421 Midnight Circle
Plano, TX
75075

Commodore Computer Club (C3)
Randy Mills
c/o Lamar Full Gospel Assembly
1200 S. Sumner
Pampa, TX
79065 806-665-3444

Gulf Coast Commodore Users Group
Lawrence Hernandez
PO Box 128
Corpus Christi, TX
78403 512-887-4577

Utah

Utah PUG
Jack Fleck
2236 Washington Blvd.
Ogden, UT
84401

Mountain Computer Society
Dave Tigner
PO Box 1154
Sandy, UT
84091

Northern Utah VIC & 64 Users Group
David Sanders
PO Box 533
Garland, UT
84312

The Commodore Users Group
Rodney Keller
652 West 700 North
Clearfield, UT
84015 801-776-3950

The Commodore Users Club
Todd Woods Kap, President
David J. Shreeve, VP
742 Taylor Ave.
Ogden, UT
84404

VIC 20 Users
Dave DeCorso
324 North, 300 West
Smithfield, UT
84335

The VIClic
Steve Graham
799 Ponderosa Dr.
Sandy, UT
84070

Virginia

VIC 20 Victims
Mike Spengel
4301 Columbia Pike #410
Arlington, VA
22204 703-920-0513

R.A.C.E. Commodore Users Group
Larry Rackow
4726 Horseman Dr.
Roanoke, VA
24019 703-362-3960

Northern VA PET Users
Bob Karpen
2045 Eakins Court
Reston, VA
22091 803-860-9116

Washington Areea C-64 (Burke)
Dick Jackson
PO Box 93
Mt. Vernon, VA
22121 703-360-6749

Peninsula Commodore 64 Users Group
Richard G. Wilmoth
124 Burnham Place
Newport News, VA
23606 804-595-7315

Dale City Commodore Users Group
Pat Sullivan
4303 Hemingway Dr.
Dale City, VA
22193 703-590-4998

Washington Area C-64 UG (McLean)
Martin Smith
c/o Kent Gardens School
7426 Eldorado St.
McLean, VA
22012 703-523-1995

PENTAF (Pentagon)
Ralph Poole
9912 Colony Rd.
Fairfax, VA
22030 703-273-1337

Arlington VICtims (20/64)
Clifton M. Gladney
Fairlington Community Center
4501 Alington Blvd.
Arlington, VA
22204 703-524-0236

Fredericksburg Area Computer Enthusiasts
Michael Parker
PO Box 324
Locust Grove, VA
22508 703-972-7195

Franconia Commodore Users Group
Mark Sowash
J. Marshall Library
6209 Rose Hill Dr.
Alexandria, VA
22310 703-971-5021

David Gray
135 Beverley Rd.
Danville, VA
24541

Norfolk Users Group
Larry Pearson
1030 West 43rd St. B-4
Norfolk, VA
23508 489-8292

Alexandria Users Group
Jeff Hendrickson
1206 Westgrove Blvd.
Alexandria, VA
22307

Commodore Users of Franklin
D. Bruce Powell
1201 N. High St.
Franklin, VA
23851 804-562-6823

Dale City Commodore Users Group
PO Box 2004
Dale City, VA
22193

NASA VIC 20 User Group
Harris Hamilton
713 York Warwick Dr.
Yorktown, VA
23692

Tidewater Commodore Users Group
Fred Monson
4917 Westgrove Rd.
Virginia Beach, VA
23455

VIC Users Group
Dick Rossignol
Rt. 2, Box 180
Lynchburg, VA
24501

Frdericksburg Computer Club
Steven Northcutt
PO Box 1011, College Station
Fredericksburg, VA
22402 703-371-4184

Capitol Area Commodore Enthusiasts
Don Swinney
P. Henry Library
2312 Tangie Vale
Vienna, VA
22180 703-938-6313

VIC Users Group
Donnie L. Thompson
1502 Harvard Rd
Richmond, VA
23226

Vermont

Burlington Area Commodore Users Group
Steve Lippert
6 Mayfair
South Burlington, VT
05402 658-4160

Washington

Central Washington Commodore Users Group
Tim McElroy
1222 S. 1st St.
Yakima, WA
98902

PET Users Group
Kenneth Tong
1800 Taylor Ave. N102
Seattle, WA
98102

Blue Mountain Commodore Users Club
Keith Rude
15 Stone St.
Walla Walla, WA
99362 509-525-5452

Central Washington Commodore Users Group
Sam Cox
PO Box 10937
Yakima, WA
98909 509-248-8193

Spokane Commodore User Group (SCUG)
Stan White
c/o N. 310 Raymond #1
Spokane, WA
99206

Fort Lewis Commodore Computer Club
Jim Litchfield
Quarters 2821-A
Fort Lewis, WA
98433 206-964-1444

Whidbey Island Commodore Computer Club
Michael D. Clark
947 N. Burroughs Ave.
Oak Harbor, WA
98277

Computer Club
John Goddard
c/o Honeywell, Inc.
5303 Shilshole Ave., NW
Seattle, WA
98107 206-789-2000

C-64 Diversity
Jill Johnston
18204 - 67th Ave., N.E
Arlington, WA
98223 206-435-4580

NW PET Users Group
Richard Bell
2565 Dexter N. 3203
Seattle, WA
98109

CBM Users Group
Rick Beaber
803 Euclid Way
Centralia, WA
98531 206-736-4085

Wisconsin

WI Asso. of VIC/64 Enthusiasts (W.A.V.E)
Annette Levandowski
PO Box 641
Waukesna, WI
53187 414-771-7016

CHIPS
Richard Kohn (E)334-2494
1017 Kilbourn Ave.
West Bend, WI
53095 414-338-1609 D

S.W.I.T.C.H.
Len Lutz
W156 N8834 Pilgrim Rd.
Menomonee Falls, WI
53051 414-255-7044

Eau Claire Area SPM 64 Users Group
John Slavsky
Rt. 5, Box 179
Eau Claire, WI
54701 715-874-5972

Waukesha Area Commodore User Group (WACUG)
Walter Sadler
256 1/2 W. Broadway
Waukesha, WI
53186 414-547-9391

Commodore 64 Software Exchange Group
E. J. Rosenberg
PO Box 224
Oregon, WI
53575

Project-20
PO Box 359
Elm Grove, WI
53122

Madison Area Commodore Users Group
John Carvin
1552 Park St.
Middleton, WI
53562 608-831-4852

C.L.U.B. 84
Jack White
6156 Douglas Ave.
Caledonia, WI
53108 414-835-4645pm

Vicky Badger Club
George Cooper
2825 Riva Ridge
Cottage Grove, WI
53527

VIC-20 & 64 User Group
Mr. Wachtl
522 West Bergen Dr.
Milwaukee, WI
53217 414-476-8125

Menomonie Area Commodore Users Group
Mike Williams
510 12th St.
Menomonie, WI
54751 715-235-4987

C.U.S.S.H.
Tim Tremmel
3614 Sovereign Dr.
Racine, WI
53406 414-554-0156

Comm Bay 64
Jeff Schwecler
2589 Haven Rd.
Green Bay, WI
54303 414-439-1619

The Eau Claire CBM64 Users Group
John Slavsky, Jr
Rt. 5, Box 179A
Eau Claire, WI
54703 715-874-5972

Milwaukee Area CBM64 Enthusiasts (M.A.C.E)
Kevin Wilde
PO Box 340
Elm Grove, WI
53122 414-259-5991

Sewpus
Theodore J. Polozynski
PO Box 21851
Milwaukee, WI
53221

Chippewa Valley Commodore 64 Users Group
Leo Lato
620 West Central St.
Chippewa Falls, WI
54729 715-723-8095

West Virginia

Marc Hutton
73 Pine Hill Estates
Kenova, WV
25530 304-453-2124

Personal Computer Club
Cam Cravens
PO Box 1301
Charleston, WV
25325

TriState Commodore Users
Marc Hutton
73 Pine Hill Estates
Kenova, WV
25530 304-453-2124

Logan Computer Club
C.R. Wilson, Jr.
PO Box 480
Logan, WV
25601

Commodore Computer Club
Chris Apperson
203 Lightner Ave.
Lewisburg, WV
24901 304-645-1150

Commodore Home Users Group - C.H.U.G.
Alice Shipley
81 Lynwood Ave.
Wheeling, WV
26003 304-242-8362

Wyoming

Commodore Users Club
Pamela Nash
c/o Video Station
670 North 3rd #B
Laramie, WY
82070 307-721-5908

Overseas

VIC Club in Helsinki
Matti Aarnio
Linnustajankj 2B7
SF-02940 ESPOO 94
Finland

Commodore Users Group
Hub Christis
HCC/Venlo, Maricollenweg 67
5971 At Grubbenvorst
Holland

Commodore 64 Club
Universita di Studi shan
V. Avigliana 13/1
10138 Torino, Italy

VIC 20 Computer Group
Lancelot Green
21 Lawrence Dr.
Kingston 8
Jamaica, West Indies

Commodore Users Club
S. K. Cha
K.PO Box 1437
Seoul, Korea

North London Hobby Computer Club
Dept. of Electronics & Communication
Engineering Polytechnic of N. London
Holloway Rd.
London N7 8DB
United Kingdom

Association Dr Usuarios Commodore
Alejandro Lopez Arechiga
Holbein 174-6 Piso
Mexico 18, D.F.

Club de Usarios Commodore
Sigma del Norte
Mol del Valle, Local 44
Garza Garcia N.L
Mexico 66220

Club Microvic
Oscar Sosa, President
Villaldama 225
Col. Chapultepec
Monterrey, N.L.
Mexico 66450

Commodore Users Group
Roger Altena
Hazel Ave.
Mount Roskill, New Zealand

Nelson VIC Users Group
Peter Archer
c/o PO Box 860
Nelson, New Zealand

c/o New Zealand Synthetic Fuels Corp.,Ltd.
E. R. Kennedy
Private Bag
New Plymouth, New Zealand

VIC Club of Norway
Nedre Bankegt 10
1750 Halden, Norway

Club de Usuarios de Commodore
Angel Fuentes Perille
c/ Guadalete no. 11-30A
Cartagena, Murcia
Spain

Croydon Microcomputer Club
Vernon Gifford
111 Selhurst
London SE25 6LH
United Kingdom

VIC-UPS Computer Users Group
Peter Prisgrove
1 Jubilee St.
South Perth 6151
West Australia

Rudi Ferrari
Kettenberg 24
D 5880 Lueden Scheid
West Germany

The Trinidad Asso. of Commodore Owners
Mark Mahannah
91 Cherry Crescent
Westmoorings/Carenage
Trinidad, West Indies

Trinidad Asso. of Computer Owners T.A.C.O
Mark Mahannah
91 Cherry Crescent
Westmoorings, Trinidad
West Indies

WA VIC-UPS (VIC 20/CBM 64 Users)
B.J. Cook
14 Glengariff Dr.
Floreat Park 6014
Western Australia

Commodore Users Club
D.A. Stagg
Postfach 5026
Salzburg, Austria

Commodore Computer Club
P.A. Stafford
c/o Syntex Corporation
PO Box F2430
Freeport, Bahamas

IEEE Standard Definitions

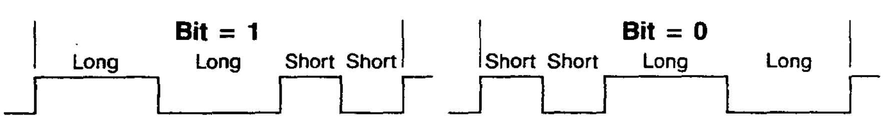
Capitalized Mnemonics represent interface states and remote messages, lowercase represent local messages received. From "IEEE Std 488-1978".

Name	Definition	Name	Definition	Name	Definition
AC	Addressed command	L or LE	Listener or extended listener	RWLS	Remote With Lockout State
ACDS	Avccept data state	LACS	Listener active status	SACS	System control active state
ACG	Addressed command group	LADS	Listener Addressed State	SCG	Secondary Command Group
ACRS	Acceptor ready state	LAG	Listener Address Group	SDC or (SDC)	Selected Device Clear
AD	Addressed	LIDS	Listener idle state	SDYS	Source delay state
AH	Acceptor handshake	LLO	Local lockout	SE	Secondary
AH1	Complete capability	LOCS	Local state	SGNS	Source generate state
AH0	No capability	lon	Listener only	SH	Source Handshake
AIDS	Acceptor idle state	LPAS	Listener Primary Addressed State	SIAS	System central interface clear active state
ANRS	Acceptor not ready state	(lpe)	Local Poll Enable	sic	Send Interface Clear
ANSI	American National Standard's Institute	LPIS	Listener Primary Idle State	SIDS	Source idle state
APRS	Affirmative Poll Response State	ltn	Listen	SIIS	System control interface clear idle state
ATN	Attention	lun	Local unlisten	SINS	System control interface clear not active state
AWNS	Acceptor Wait for New cycle State	LWLS	Local With Lockout State	SIWS	Source Idle Wait State
C	Controller	M	Multiline	SNAS	System control not active state
CACS	Controller addressed state	MLA or (MLA)	My Listen Address	SPAS	Serial Poll Active State
CADS	Controller idle state	MSA or (MSA)	My Secondary Address	SPD	Serial Poll Disable
CAWS	Controller active wait state	MTA or (MTA)	My Talk Address	SPE	Serial Poll Enable
CIDS	Controller idle state	nba	New Byte Available	SPIS	Serial Poll Idle State
CPPS	Controller parallel poll state	NDAC	Not Data Accepted	SPMS	Serial Poll Mode State
CPWS	Controller parallel poll wait state	NPRS	Negative Poll Response State	SR	Service Request
CSBS	Controller standby state	NRFD	Not Ready For Data	SRAS	System control remote enable active state
CSNS	Controller service not requested state	NUL	Null byte	sre	Send Remote Enable
CSRS	Controller service requested state	OSA	Other Seconday Address	SRIS	System control remote enable idle state
CSWS	Controller synchronous wait state	OTA	Other Talk Address	SRNS	System control remote enable not active state
CTRS	Controller transfer state	PACS	Parallel poll addressed to configure state	SRQ	Service request
DAB	Data byte	PCG	Primary Command Group	SRQS	Service request state
DAC	Data accepted	POFS	Power off	ST	Status
DAV	Controller Data valid	pon	Power on	STB	Status Byte
DC	Device clear	PP	Parallel Poll	STRS	Source Transfer State
DCAS	Device clear active state	PPAS	Parallel Poll Active State	SWNS	Source wait for new cycle state
DCIS	Device clear idle state	PPC	Parallel Poll configure	T or (TE)	Talker or extended talker
DCL	Device clear	PPD or (PPD)	Parallel Poll Disable	T	Active true
DD	Device Dependent	PPE or (PPE)	Parallel Poll Enable	(T)	Passive True
DIO	Data input	PPIS	Parallel Poll Idle State	TACS	Talker active state
DT	Device trigger	PPR	Parallel Poll Response	TADS	Talker addressed state
DTAS	Device Trigger Active State	PPSS	Parallel Poll Standby State	TAG	Talk Address Group
DTIS	Device trigger state	PPU	Parallel Poll Unconfigure	tca	Take Control Asynchronously
END	End	PUCS	Parallel poll unaddressed to configure state	tcs	Take Control Synchronously
EOI	End Or Identity	rdy	Ready (for next message)	TCT or (TCT)	Take control
EOS	End Of String	REMS	Remote state	TIDS	Talker idle state
F	Active false	REN	Remote enable	ton	Talk only
(F)	Passive False	RFD	Ready For Data	TPAS	Talker Primary Addressed State
GET	Group Execute Trigger	RL	Remote Local	U	Uniline message
GTL	Go To Local	rpp	Request Parallel Poll	UC	Universal Command
gts	Go To Standby	RQS	Request service	UCG	Universal Command Group
IDY	Identify	rsc	Request System Control	UNL	Unlisten
IFC	Interface clear	rsv	Request service	UNT	Untalk
ist	Individual status	rtl	Return To Local		

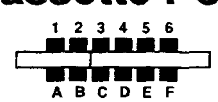
Tape Recording Format

Leader = 50 cycles of shorts
Mark = 342 micro seconds of 1.46 KHz half cycle
Short = 182 micro seconds of 2.75 KHz half cycle
Long = 262 micro seconds of 1.91 KHz half cycle





Cassette Port

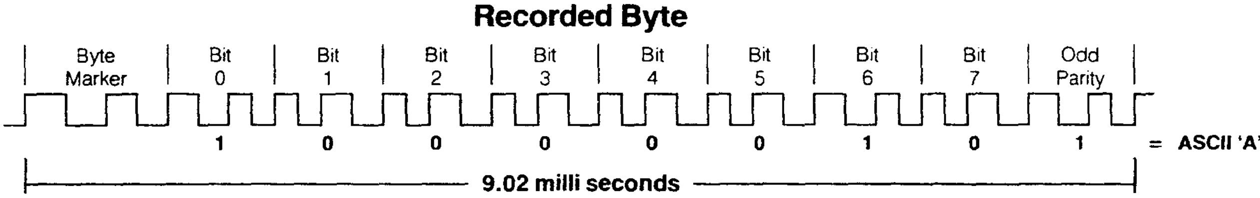


(Edge-on view)

Pin#	Name	Description
A-1	GND	Digital Ground
B-2	+5V	+5 Volts to operate cassette circuitry only
C-3	Motor	Computer controlled +6V for cassette motor
D-4	Read	Read line from cassette
E-5	Write	Write line cassette
F-6	Sense	Monitors closure of any locking type cassette switch

Note: Upper and Lower cassette pins are shorted

Recorded Byte



Tape File Format

Program File

Leader	Header (192 Bytes)	Repeated Header	Program	Repeated Program	End (192 Bytes)	Repeated End

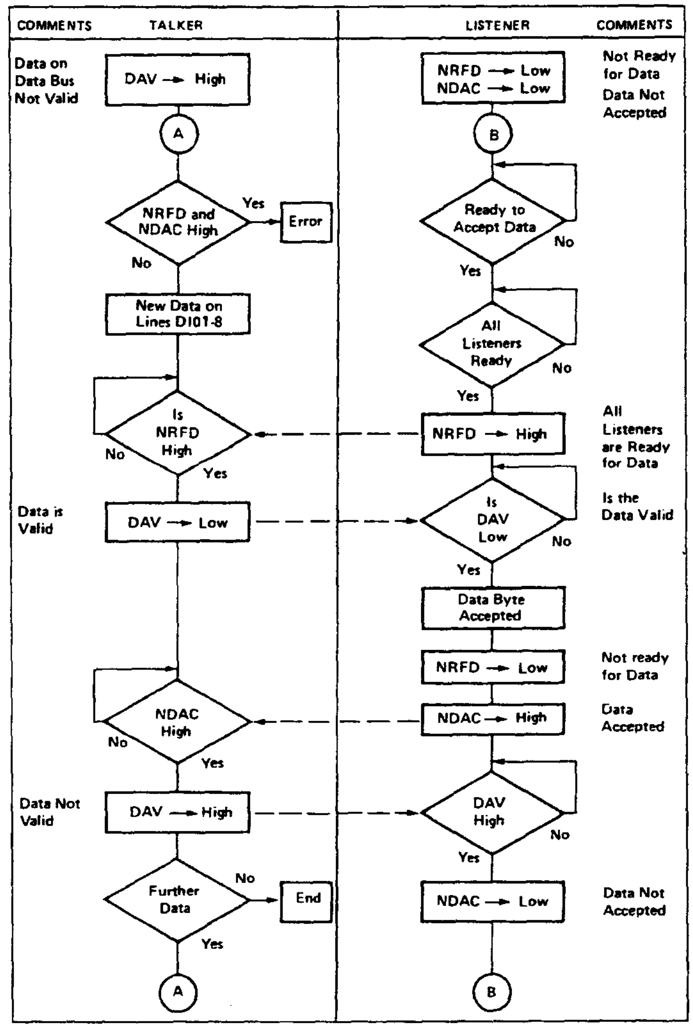
Data File

Leader	Header (192 Bytes)	Repeated Header	Data Block (192 Bytes)	Repeated Data Block	Data Block	Repeated Data Block (etc. to end of file)	End (192 Bytes)	Repeated End

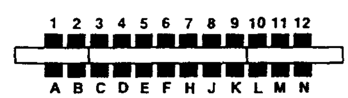
IEEE 488 Bus Signals

Manager	ATN	Attention	The controller (PET/CBM/B) sets this signal low while it is sending commands on the data bus. When ATN is low, only peripheral addresses and control messages are on the data bus. When ATN is high, only previously assigned devices can transfer data.
Transfer	DAV	Data Valid	When DAV is low, this signifies that data is valid on data bus.
Manager	EOI	End or Identify	When the last byte of data is being transferred, the talker has the option of setting EOI low. The controller always sets EOI low while the last data byte is being transferred from the controller.
Manager	IFC	Interface Clear	The controller sends its internal reset signal as IFC low (true) to initialize all devices to the idle state. When the controller is switched on or reset, IFC goes low for about 100 milliseconds.
Transfer	NDAC	Data Not Accepted	This signal is held low (true) by the listener while reading. When the data byte has been read, the listener sets NDAC high. This signals the talker that data has been accepted.
Transfer	NRFD	Not Ready for Data	When NRFD is low (true), one or more listeners are not ready for the next byte of data. When all devices are ready, NRFD goes high.
Manager	SRQ	Service Request	Not implemented in BASIC, but available to the user.
Manager	REN	Remote Enable	REN is held low by the bus controller. The PET/CBM has a pin grounded that keeps REN permanently low.
Data	D101-8	Data Input/Output Lines 1–8	These signals represent the bits of information on the data bus. When a D10 signal is low, it represents 1 and when high 0.
General	GND	Ground	Ground connections: There are six control and management signal ground returns, one data signal ground return and one chassis shield ground lead.

IEEE Byte Transfer Sequence



IEEE Port Pinouts

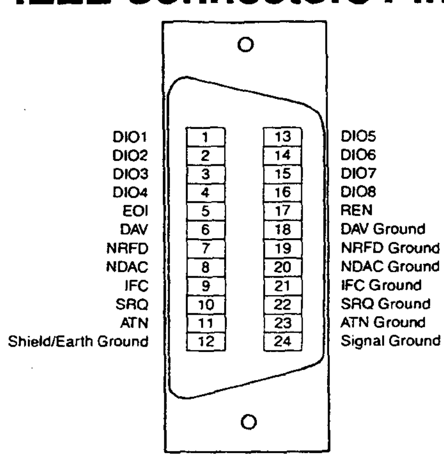


(Edge-on view)

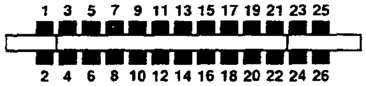
Pin #	Pin#*	Mnemonic	Definition
1	1	DIO1	Data Input/Output Line #1
2	2	DIO2	Data Input/Output Line #2
3	3	DIO3	Data Input/Output Line #3
4	4	DIO4	Data Input/Output Line #4
5	5	EOI	End or Identify
6	6	DAV	Data Valid
7	7	NRFD	Not Ready For Data
8	8	NDAC	Data not Accepted
9	9	IFC	Interface Clear
10	10	SRQ	Service Request
11	11	ATN	Attention
12	12	GND	Chassis Ground (IEEE cable shield)
A	13	DIO5	Data Input/Output Line #5
B	14	DIO6	Data Input/Output Line #6
C	15	DIO7	Data Input/Output Line #7
D	16	DIO8	Data Input/Output Line #8
E	17	REN	Remote Enable
F	18	GND	DAV Ground
H	19	GND	NRFD Ground
J	20	GND	NDAC Ground
K	21	GND	IFC Ground
L	22	GND	SRQ Ground
M	23	GND	ATN Ground
N	24	GND	Data Ground (DIO1–8)

* Pin Numbers for Standard IEEE Cable Connector

IEEE Connectors Pins



PET/CBM User Port



(Edge-on view)

Pin#	Function	Description
1	Ground	System Ground
2	TV Video	Video Out for external displays
3	SRQ	Connected to IEEE SRQ
4	EOI	Connected to IEEE EOI
5	Diag Sense	Held low causes power up to Diagnostic routines
6	READ 1	Connected to cassette 1 read line
7	READ 2	Connected to cassette 2 read line
8	Write	Diagnostic tape write verify
9	Vert	TV Vertical for external displays
10	Horiz	TV Horizontal for external displays
11	GND	
12	GND	
A	GND	
B	CA1	Edge sensitive input of 6522 VIA
C	PB0	PB0-7 are independently programmable
D	PB1	for Input or Output
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	CB2	Special I/O pin of VIA
N	GND	Digital Ground

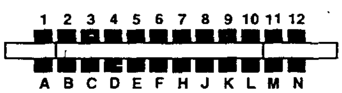
6522 Registers

2 8-Bit I/O Ports, 4 Control Lines, 2 16-Bit Counter/Timers, 1 8-Bit Shift Register

Reg#	Register Function
0	I/O Port B Data register
1	I/O Port A Data register, with handshaking
2	I/O Port B Data Direction
3	I/O Port A Data Direction
4	Read: Timer 1 Counter low. Resets T1 Int. Flag (IFR Bit6) Write: Timer 1 Latch low. T1 Latch low xferred to T1 Counter low on writin Reg 5
5	Read: Timer 1 Counter high. Write: Timer 1 Latch high. Latch high transferred to T1 on writing
6	Write: Timer 1 Latch low. Contents transferred to Reg 4 Read: Timer 1 Latch low. Does not reset T1 Int. Flag
7	Write: Timer 1 Latch high. Start up value, no transfer Read: Timer 1 Latch high.
8	Write: Timer 2 low. Read: Timer 2 low.
9	Write: Timer 2 high. Transfers T2 Latch low to T2 Counter low. Resets T2 Int. Flag (IFR Bit5)
10	Serial I/O shift register. Shift OUT: Bit 7 first out, then rotated to Bit 0 Shift IN: Bit 0 loaded first, rotated towards Bit 7
11	Auxhiliary Control register
12	Peripheral Control register
13	Interrupt Flag Register (IFR)
14	Interrupt Enable Register (IER)
15	I/O Port A Data, no handshaking

DDRA/B: Bit = 0 Input, Bit = 1 Output (Remember: NOT I/O)

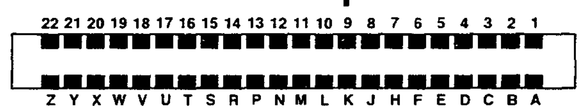
Commodore 64 User Port



(Edge-on view)

Pin#	Function	Description
1	Ground	System Ground
2	+5V	(100 ma maximum)
3	RESET	Cold Start. Memory is NOT destroyed
4	CNT1	Serial Port counter from CIA #1
5	SP1	Serial Port from CIA #1
6	CNT2	Serial Port counter from CIA #2
7	SP2	Serial Port from CIA #2
8	PC2	Handshaking line from CIA #2
9	Serial ATN	Connected to Serial Bus ATN Line
10	9 VAC + Phase	Transformer output (50 ma. maximum)
11	9 VAC -Phase	Transformer output (50 ma. maximum)
12	GND	
A	GND	
B	FLAG2	
C	PB0	PB0-7 are independently programmable
D	PB1	for Input or Output
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	PA2	Special I/O pin of CIA
N	GND	

Commodore 64 Expansion Port



(End view)

Pin#	Name	Description
1	GND	System Ground.
2	+5 VDC	Total User Port and Cartridge devices
3	+5 VDC	can draw no more than 450ma.
4	IRQ	Interupt Request line to 6510 (active low).
5	R/W	Read/Write.
6	Dot Clock	8.18 MHz video dot clock.
7	I/O 1	I/O Block 1 @ $DE00-$DEFF (active low) unbuffered I/O.
8	GAME	Active low TTL input.
9	EXROM	Active low TTL input.
10	I/O 2	I/O Block 2 @ $DF00-$DFFF (active low) buffered TTL output.
11	ROM L	8K decoded RAM/ROM block @ $8000 (active low) buffered TTL output.
12	BA	Bus Available signal from the VIC II chip - unbuffered - 1 is maximum load.
13	DMA	Direct Memory Access request line (active low input) is TTL input.
14	D7	Data bus bit 7 *
15	D6	Data bus bit 6 *
16	D5	Data bus bit 5 *
17	D4	Data bus bit 4 *
18	D3	Data bus bit 3 *
19	D2	Data bus bit 2 *
20	D1	Data bus bit 1 *
21	D0	Data bus bit 0 *
21	GND	System ground.
A	GND	System Ground
B	ROM H	8K decoded RAM/ROM Block @ $E000 buffered.
C	RESET	6510 RESET pin (active low) buffered TTL out/unbuffered in.
D	NMI	6510 Non-Maskable Interrupt (active low) buffered TTL out, unbuffered in.
E	Φ2	Phase 2 system clock.
F	A15	Address bus bit 15 *
H	A14	Address bus bit 14 *
J	A13	Address bus bit 13 *
K	A12	Address bus bit 12 *
L	A11	Address bus bit 11 *
M	A10	Address bus bit 10 *
N	A9	Address bus bit 9 *
P	A8	Address bus bit 8 *
R	A7	Address bus bit 7 *
S	A6	Address bus bit 6 *
T	A5	Address bus bit 5 *
U	A4	Address bus bit 4 *
V	A3	Address bus bit 3 *
W	A2	Address bus bit 2 *
X	A1	Address bus bit 1 *
Y	A0	Address bus bit 0 *
Z	GND	System Ground

C64 / VIC 20 Keyboard Matrix

ROW	Column (bit in location 56321)							
	7	6	5	4	3	2	1	0
$FE	dn	F5	F3	F1	F7	rt	rtrn	DEL
$FD	l. shft	E	S	Z	4	A	W	3
$FB	X	T	F	C	6	D	R	5
$F7	V	U	H	B	8	G	Y	7
$EF	N	O	K	M	0	J	I	9
$DF	,	@	:	.	–	L	P	+
$BF	/	↑	=	r.shf	HOME	;	*	£
$7F	STOP	Q	C=	SPACE	2	CTRL	←	1

Notes:
1) The Shift Lock Key is connected to the left shift key.
2) The RESTORE Key is not part of the keyboard matrix, but is directly wired to generate an NMI Interrupt when struck.

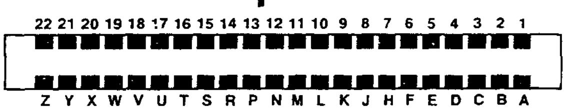
VIC 20 User Port



(Edge-on view)

Pin#	Name	Description
1	Ground	System Ground
2	+5V	(100 ma maximum)
3	RESET	Cold Start. Memory is destroyed
4	JOY 0	Joystick Switch 0
5	JOY 1	Joystick Switch 1
6	JOY 2	Joystick Switch 2
7	PEN	Light Pen Input. Also Joystick Fire Button
8	SENSE	Cassette Switch sense line
9	Serial ATN	Connected to Serial Bus ATN Line
10	9 VAC +Phase	Transformer output (50 ma. maximum)
11	GND	
12	GND	
A	GND	
B	CB1	
C	PB0	PB0-7 are independently programmable
D	PB1	for Input or Output
E	PB2	
F	PB3	
H	PB4	
J	PB5	
K	PB6	
L	PB7	
M	CB2	Special I/O pin of VIA
N	GND	

VIC 20 Expansion Port

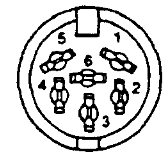


(End view)

Pin#	Name	Description
1	GND	System ground
2	CD0	Data bus bit 0 *
3	CD1	Data bus bit 1 *
4	CD2	Data bus bit 2 *
5	CD3	Data bus bit 3 *
6	CD4	Data bus bit 4 *
7	CD5	Data bus bit 5 *
8	CD6	Data bus bit 6 *
9	CD7	Data bus bit 7 *
10	BLK1	8k decoded RAM/ROM block 1 @ $2000 (active low)
11	BLK2	8k decoded RAM/ROM block 2 @ $4000 (active low)
12	BLK3	8k decoded RAM/ROM block 3 @ $6000 (active low)
13	BLK5	8k decoded ROM block 5 @ $A000 (active low)
14	RAM1	1k decoded RAM block @ $0400 (active low)
15	RAM2	1k decoded RAM block @ $0800 (active low)
16	RAM3	1k decoded RAM block @ $0C00 (active low)
17	V R/W	Read/Write line from VIC Chip (high-read, low-write)
18	C R/W	Read/Write line from CPU (high-read, low-write)
19	IRQ	Interupt Request line to 6502 (active low)
20	NC	
21	+5v	
22	GND	
A	GND	
b	CA0	Address bus bit 0 *
C	CA1	Address bus bit 1 *
D	CA2	Address bus bit 2 *
E	CA3	Address bus bit 3 *
F	CA4	Address bus bit 4 *
H	CA5	Address bus bit 5 *
J	CA6	Address bus bit 6 *
K	CA7	Address bus bit 7 *
L	CA8	Address bus bit 8 *
M	CA9	Address bus bit 9 *
N	CA10	Address bus bit 10 *
P	CA11	Address bus bit 11 *
R	CA12	Address bus bit 12 *
S	CA13	Address bus bit 13 *
T	I/O 2	I/O block 2 (located at $9600)
U	I/O 3	I/O block 3 (located at $9C00)
V	Φ02	Phase 2 system clock
W	NMI	6502 Non-Maskable Interrupt (active low)
X	RESET	6502 Reset pin (active low)
Y	NC	
Z	GND	

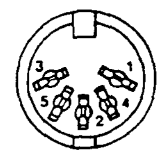
* = Unbuffered, 1 low power Schottky TTL load max.

VIC 20 / Commodore 64 Serial Port



Pin#	Name	Description
1	SRQ	Serial SRQ in (active low)
2	GND	System Ground
3	ATN	Serial ATN In/Out
4	CLK	Serial Clock In/Out
5	DATA	Serial Data In/Out
6	RESET	Resets all devices on Serial bus (active low)

VIC 20 Audio/Video Port



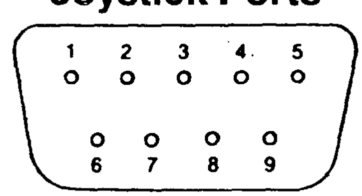
Pin#	Name	Description	Colour
1	+5V	10 ma. maximum	Red
2	GND	System Ground	-
3	AUD	Audio Out	Grey
4	VID L	Video Low	Black
5	VID H	Video High	White

Colour refers to Radio Shack Part# 42-2394

Commodore 64 Audio/Video Port

Pin#	Name	Description
1	LUM	Luminance
2	GND	System Ground
3	AUD	Audio Out
4	COMP	Composite Video
5	JACK	Audio In
6	CHR	Chroma out
7	N/C	No connection
8	N/C	No connection

VIC 20 / Commodore 64 Joystick Ports



Pin#	Name	Description
1	JOY 0	
2	JOY 1	
3	JOY 2	
4	JOY 3	
5	POT Y	
6	FIRE	Also the Light Pen input. (C64 port 1 only)
7	+5V	100 ma. maximum
8	GND	System Ground
9	POT X	

Note: See Memory Map for reading Joystick Ports

6520 PIA Registers

2 8-Bit I/O Ports, 4 Control Lines.
Control Register Bit 2 is used to select Data or Direction Registers

Reg#	**CRA Bit 2 =**	**Register Function**
0	0	I/O Port A Data Direction Register (DDRA)
0	1	Peripheral I/O Port A Data register (PA)
1		Control Register A (CRA)
Reg#	**CRB Bit 2 =**	**Register Function**
2	0	I/O Port B Data Direction Register (DDRB)
2	1	Peripheral I/O Port B Data register (PB)
3		Control Register B (CRB)

DDRA/B: Bit = 0 Input, Bit = 1 Output (Remember: NOT I/O)

PIA Control Registers

Control Register A (Chip Address + 1)

7	6	5	4	3	2	1	0
IRQ 1	IRQ 2	CA2 Control			DDRA Access	CA1 Control	

Bit 1	Bit 0	
0	0	IRQ1 = 1 on CA1 going low, $\overline{IRQ}$ disabled
0	1	IRQ1 = 1 on CA1 going low, $\overline{IRQ}$ enabled
1	0	IRQ1 = 1 on CA1 going high, $\overline{IRQ}$ disabled
1	1	IRQ1 = 1 on CA1 going high, $\overline{IRQ}$ enabled
		IRQ1 cleared by reading Peripheral I/O Register A

Bit 2	
0	Chip Address + 0 = DDRA
1	Chip Address + 0 = Peripheral I/O Port A

Bit 5	Bit 4	Bit 3	
0	0	0	IRQ2 = 1 on CA2 going low, $\overline{IRQ}$ disabled
0	0	1	IRQ2 = 1 on CA2 going low, $\overline{IRQ}$ enabled
0	1	0	IRQ2 = 1 on CA2 going high, $\overline{IRQ}$ disabled
0	1	1	IRQ2 = 1 on CA2 going high, $\overline{IRQ}$ enabled
			IRQ2 cleared by reading Peripheral I/O Register A
1	0	0	CA2 set on IRQ1 = 1 (CA1 transition)
1	0	0	CA2 cleared after PA read
1	0	1	CA2 set on every Φ2 negative edge
1	0	1	CA2 goes low for 1 Φ2 cycle after PA read
1	1	0	Manual control: CA2 low
1	1	1	Manual control: CA2 high

Bit 7	Bit 6	
X	X	IRQ Flags Set by external events
X	X	IRQ Flags Cleared by reading I/O ports

Control Register B (Chip Address + 3)

7	6	5	4	3	2	1	0
IRQ 1	IRQ 2	CB2 Control			DDRB Access	CB1 Control	

Bit 1	Bit 0	
0	0	IRQ1 = 1 on CB1 going low, $\overline{IRQ}$ disabled
0	1	IRQ1 = 1 on CB1 going low, $\overline{IRQ}$ enabled
1	0	IRQ1 = 1 on CB1 going high, $\overline{IRQ}$ disabled
1	1	IRQ1 = 1 on CB1 going high, $\overline{IRQ}$ enabled
		IRQ1 cleared by reading Peripheral I/O Register B

Bit 2	
0	Chip Address + 2 = DDRB
1	Chip Address + 2 = Peripheral I/O Port B

Bit 5	Bit 4	Bit 3	
0	0	0	IRQ2 = 1 on CB2 going low, $\overline{IRQ}$ disabled
0	0	1	IRQ2 = 1 on CB2 going low, $\overline{IRQ}$ enabled
0	1	0	IRQ2 = 1 on CB2 going high, $\overline{IRQ}$ disabled
0	1	1	IRQ2 = 1 on CB2 going high, $\overline{IRQ}$ enabled
			IRQ2 cleared by reading Peripheral I/O Register B
1	0	0	CB2 set on IRQ1 = 1 (CB1 transition)
1	0	0	CB2 cleared after PB write
1	0	1	CB2 set on every Φ2 negative edge
1	0	1	CB2 goes low for 1 Φ2 cycle after PB write
1	1	0	Manual control: CB2 low
1	1	1	Manual control: CB2 high

Bit 7	Bit 6	
X	X	IRQ Flags Set by external events
X	X	IRQ Flags Cleared by reading I/O ports

6522 VIA Control Registers

Auxillary Control Register (Chip Address + 11)

7	6	5	4	3	2	1	0
Timer 1 Ctrl		T2 Control	Shift Reg Control			Latch Ctrl	

Bit 1	Bit 0	
0	0	PA Latch disabled, PB Latch disabled
1	1	PA Latch enabled, PB Latch enabled

Bit 4	Bit 3	Bit 2	
0	0	0	Shift Register disabled
0	0	1	Shift IN: shift rate controlled by Timer 2
0	1	0	Shift IN: shift rate controlled by Φ2
0	1	1	Shift IN: shift rate controlled by External Clock source
1	0	0	Shift OUT: Free-Running Mode, rate controlled by Timer 2
1	0	1	Shift OUT: rate controlled by Timer 2
1	1	0	Shift OUT: rate controlled by Φ2
1	1	1	Shift OUT: rate controlled by External Clock source

Bit 5	
0	Decrement Counter 2 at Φ2 clock rate (in one-shot mode)
1	Decrement Counter 2 on pulses from PB6

Bit 6	
0	One-Shot Mode
1	Free-Running Mode

Bit 7	
0	PB7 disabled
1	PB7 enabled

Interrupt Flag Register (Chip Address + 13)

7	6	5	4	3	2	1	0
IRQ	T1	T2	CB1	CB2	SR	CA1	CA2

Bit	Flag Set	Flag Cleared
CA2	Transition at CA2	Reading/Writing I/O Port A
CA1	Transition at CA1	Reading/Writing I/O Port A
SR	8 Bits Shifted IN/OUT	Reading/Writing Shift Reg
CB2	Transition at CB2	Reading/Writing I/O Port B
CB1	Transition at CB1	Reading/Writing I/O Port B
T2	Timer 2 Timeout	Reading T2 low / Writing T2 High
T1	Timer 1 Timeout	Reading T1 low / Writing T1 High
IRQ	Interrupt Occuring	Clearing any interrupt

Interrupt Enable Register (Chip Address + 14)

7	6	5	4	3	2	1	0
S/C	T1	T2	CB1	CB2	SR	CA1	CA2

T1	T2	CB1	CB2	SR	CA1	CA2	
0	0	0	0	0	0	0	Interrupt Disabled
1	1	1	1	1	1	1	Interrupt Enabled

S/C: Set Enable Flag: write 1 OR'd with Flag Bit n = 1
Clear Enable Flag: write 0 OR'd with Flag Bit n = 1

Peripheral Control Register (Chip Address + 12)

7	6	5	4	3	2	1	0
CB2 Control			CB1 IRQ Control	CA2 Control			CA1 IRQ Control

Bit 0	
0	Interrupt Flag Reg Bit1 = 1 on CA1 going low
1	Interrupt Flag Reg Bit1 = 1 on CA1 going high
	Interrupt Flag Reg Bit1 cleared by reading I/O Port A

Bit 3	Bit 2	Bit 1	
0	0	0	Input Mode: IFR Bit0 = 1 on CA2 going low (IFR Bit0 cleared by read/write of I/O Port A)
0	0	1	Independent Int. Input Mode: IFR Bit0 = 1 on CA2 going low (IFR Bit0 is not cleared by read/write of I/O Port A)
0	1	0	Input Mode: IFR Bit0 = 1 on CA2 going high (IFR Bit0 cleared by read/write of I/O Port A)
0	1	1	Independent Int. Input Mode: IFR Bit0 = 1 on CA2 going high (IFR Bit0 is not cleared by read/write of I/O Port A)
1	0	0	Output Mode w/Handshaking: CA2 goes low on reading/writing I/O Port A (CA2 goes high on pulse from CA1)
1	0	1	Pulse Output Mode: CA2 goes low for one Φ2 cycle on reading/writing I/O Port A
1	1	0	Manual Output: CA2 set low
1	1	1	Manual Output: CA2 set high

Bit 4	
0	Interrupt Flag Reg Bit4 = 1 on CB1 going low
1	Interrupt Flag Reg Bit4 = 1 on CB1 going high
	Interrupt Flag Reg Bit4 cleared by reading I/O Port B

Bit 7	Bit 6	Bit 5	
0	0	0	Interrupt Input Mode: IFR Bit3 = 1 on CB2 going low (IFR Bit3 cleared by reading/writing I/O Port B)
0	0	1	Independent Int. Input Mode: IFR Bit3 = 1 on CB2 going low (IFR Bit3 is not cleared by reading/writing I/O Port B)
0	1	0	Input Mode: IFR Bit3 = 1 on CB2 going high (IFR Bit3 cleared by reading/writing I/O Port A)
0	1	1	Independent Int. Input Mode: IFR Bit3 = 1 on CB2 going high (IFR Bit3 is not cleared by reading/writing I/O Port A)
1	0	0	Output Mode w/Handshaking: CB2 goes low on reading/writing I/O Port A (CB2 goes high on pulse from CB1)
1	0	1	Pulse Output Mode: CB2 goes low for one Φ2 cycle on reading/writing I/O Port A
1	1	0	Manual Output: CB2 set low
1	1	1	Manual Output: CB2 set high

6526 CIA Registers

Control Register A (Chip Address + 14)

7	6	5	4	3	2	1	0
TOD freq	SP mode	Input Mode	Load	Run Mode	Output Mode	PB6 Control	Start/Stop

Bit	Value	Function
0	0	Stop Timer A
0	1	Start Timer A, Reset on timeout in One-Shot mode
1	0	PB6 Normal operation
1	1	PB6 = Output of Timer A
2	0	Port B output toggled
2	1	Port B positive pulse at Φ2 clock rate
3	0	Free-Running Mode
3	1	One-Shot Mode
4	0	no effect
4	1	Transfer A Latches to A Counters, timer running or stopped
5	0	Timer A counts at Φ2 clock rate
5	1	Timer A counts on positive transitions from pin 40 (CNT)
6	0	Serial Port (pin 39) input at external clock rate (pin 40, CNT)
6	1	Serial Port output at ½ TA timeout rate minus line loading (max Φ2 ÷ 2)
7	0	TOD at 60 Hz (apply 60Hz to pin 19)
7	1	TOD at 50 Hz (apply 50Hz to pin 19)

Reg#	Register Function
1	I/O Port A Data register
0	I/O Port B Data register
3	I/O Port A Data Direction
2	I/O Port B Data Direction
4	Read: Timer A Counter low. Resets TA Int. Flag (ICR Bit0) Write: Timer A Latch low. TA Latch low xferred to TA Counter low on writing Reg 5
5	Read: Timer A Counter high. Write: Timer A Latch high. Latch high transferred to TA on writing
6	Read: Timer B Counter low. Resets TB Int. Flag (ICR Bit1) Write: Timer B Latch low. TB Latch low xferred to TA Counter low on writing Reg 7
7	Read: Timer B Counter high. Write: Timer B Latch high. Latch high transferred to TB on writing

Interrupt Control DATA Register (read) (Chip Address + 13)

7	6	5	4	3	2	1	0
IRQ	0	0	FLG	Serial Port	Alrm	TB	TA

Bit	Flag Set	Flag Cleared
0 (TA)	Timer A Timeout	Reading ICR*
1 (TB)	Timer B Timeout	Reading ICR*
2 (Alrm)	TOD = Alarm Settings	Reading ICR*
3 (Serial Port)	8 shifts of Serial Port (IN or OUT)	Reading ICR*
4 (FLG)	FLAG pin grounded (pin 24)	Reading ICR*
7 (IRQ)	Interrupt Occuring	Reading ICR*

* User responsible for preserving flags in case of multiple interrupts

Interrupt Control MASK Register (write) (Chip Address + 13)

7	6	5	4	3	2	1	0
S/C	X	X	FLG	Serial Port	Alrm	TB	TA

FLG	Serial Port	Alrm	TB	TA	
0	0	0	0	0	Interrupt Disabled, X = unused
1	1	1	1	1	Interrupt Enabled

Bit 7 (S/C):
Set Enable Flag: write 1 OR'd with Flag Bit n = 1
Clear Enable Flag: write 0 OR'd with Flag Bit n = 1

Control Register B (Chip Address + 15)

7	6	5	4	3	2	1	0
TOD/Alarm Access	Input Mode		Load	Run Mode	Output Mode	PB7 Control	Start/Stop

Bit	Value	Function
0	0	Stop Timer B
0	1	Start Timer B, Reset on timeout in One-Shot mode
1	0	PB7 Normal operation
1	1	PB7 = Output of Timer B
2	0	Port B output toggled
2	1	Port B positive pulse at Φ2 clock rate
3	0	Free-Running Mode
3	1	One-Shot Mode
4	0	no effect
4	1	Transfer B Latches to B Counters, timer running or stopped
7	0	Writing to TOD registers sets Clock values
7	1	Writing to TOD registers sets Alarm values

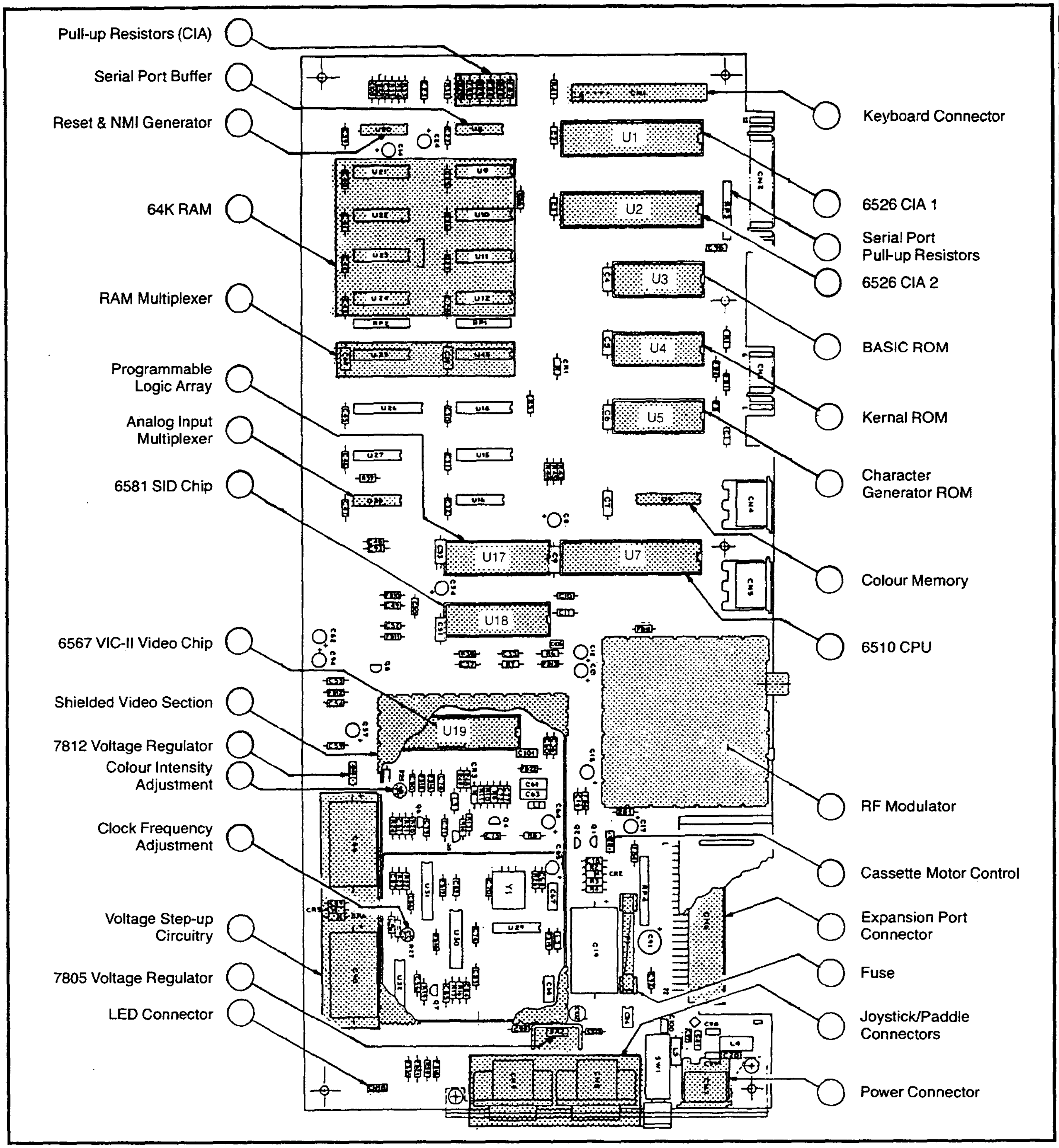
Bit 6	Bit 5	Function
0	0	Timer B counts at Φ2 clock rate
0	1	Timer B counts on positive transitions from pin 40 (CNT)
1	0	Timer B counts Timer A timeouts
1	1	Timer B counts Timer A timeouts while pin 40 (CNT) is high

	Time Of Day Clock, Read or Write — nu = not used
8	CRB Bit7 = 0: TOD 10ths. Bits 0-3 hold 10ths of seconds in BCD (bits 4-7 nu). Writing Reg 8 starts clock.
8	CRB Bit7 = 1: Alarm 10ths, same format, write only.
9	CRB Bit7 = 0: TOD Secs in BCD (Bits 0-3 + Bits 4-6 × 10, B7 nu)
9	CRB Bit7 = 1: Alarm Seconds, same format, write only.
10	CRB Bit7 = 0: TOD Mins in BCD (Bits 0-3 + Bits 4-6 × 10, B7 nu)
10	CRB Bit7 = 1: Alarm Minutes, same format, write only.
11	CRB Bit7 = 0: TOD Hours in BCD (Bits 0-3 + Bit 4 × 10, Bits 5 and 6 nu. Bit 7 = AM/PM) Reading Reg 11 latches TOD values, but clock continues. Reading Reg 8 (10ths) disables latch.
11	CRB Bit7 = 1: Alarm Hours, same format, write only.
12	Serial Data Reg. Shift OUT: Bit7 first out. Shift IN: Bit0 first in, shifted towards Bit7.
13	Interrupt Control Register (ICR)
14	Control Register A (CRA)
15	Control Register B (CRB)

DDRA/B: Bit = 0 Input, Bit = 1 Output (Remember: NOT I/O)

Commodore 64 Board Layout

At least 3 circuit boards exist, but differences are minor in most cases.



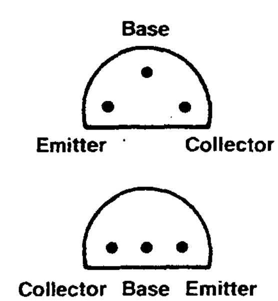
Resistor Colour Codes



1st Band: 1st Digit
2nd Band: 2nd Digit
3rd Band: Multiplier (# of Zeros)
4th Band: Tolerance

Colour	Value	"Remember:"	Fractional Multipliers	
Black	0	Bad	**Colour**	**Multiply by:**
Brown	1	Boys	Gold	0.10
Red	2	Rape	Silver	0.01
Orange	3	Our	**Tolerance Percents**	
Yellow	4	Young	No Band	±20%
Green	5	Girls	Silver	±10%
Blue	6	But	Gold	±5%
Violet	7	Violet		
Grey	8	Gives		
White	9	Willingly		

Transistor Leads



ACIA / VIC 20 / Commodore 64 / B / +4 RS 232 Control

Features not common to all machines are so noted.

OPEN LF, 2, SA, CHR$(Control Register [7 6 5 4 3 2 1 0]) + CHR$(Command Register [7 6 5 4 3 2 1 0])

B Series: + CHR$(0) + CHR$(0) not used but necessary

SA	B Series:
1	Open Output Channel
2	Open Input Channel
3	Open Input/Output Channel
129	Output Channel, Convert CBM to ASCII
130	Input Channel, Convert ASCII to CBM
131	Input/Output, Convert ASCII⇌CBM

ACIA / VIC 20 / C64 / B / +4 RS 232 Status

7 6 5 4 3 2 1 0 — ST: Status Variable = Status Register

Bit	Meaning
0	1 = Parity Error
1	1 = Framing Error
2	1 = Receiver Buffer Overrun
3	ACIA: 1 = Receiver Register Full; VIC/64: 0 = Receiver Buffer Empty
4	ACIA: 1 = Transmitter Register Empty; VIC/64: 1 = CTS Signal Missing
5	1 = Carrier Detected
6	1 = Data Set Not Ready
7	1 = Interrupt Has Occured

Control Register

Baud (bits 3–0)

Bit 3	Bit 2	Bit 1	Bit 0	Baud
0	0	0	0	User*
0	0	0	1	50
0	0	1	0	75
0	0	1	1	110
0	1	0	0	134.5
0	1	0	1	150
0	1	1	0	300
0	1	1	1	600
1	0	0	0	1200
1	0	0	1	2400
1	0	1	0	2400
1	0	1	1	3600*
1	1	0	0	4800*
1	1	0	1	7200*
1	1	1	0	9600*
1	1	1	1	19200*

* VIC/64: not implemented
B/+4: User = 1/16 External

RCVR Clock (bit 4)

VIC/64		ACIA/B/+4
X	Not Used	
	0	External
	1	Internal

Word Length (bits 6–5)

Bit 6	Bit 5	
0	0	8 Bits
0	1	7 Bits
1	0	6 Bits
1	1	5 Bits

Stop Bits (bit 7)

Bit 7	
0	1 Stop Bit
1	2 Stop Bits

Command Register

B Series

X	X	X	X	Not Used

Bit 0

	VIC/64 Handshake	ACIA/+4 Data Terminal Ready
0	3 Line	Disable Rcv/Xmit (DTR high)
1	X Line	Enable Rcv/Xmit (DTR low)

VIC/64

X	X	X	Not Used

ACIA and +4 Receiver Interrupt (bit 1)

0	Enable IRQ from Status Reg Bit 0
1	Disable IRQ Interrupt

ACIA and +4 Transmitter Controls (bits 3–2)

Bit 3	Bit 2	Transmit Interrupt	RTS Level	Other
0	0	Disabled	High	—
0	1	Enabled	Low	—
1	0	Disabled	Low	—
1	1	Disabled	Low	Transmit BRK

Duplex (bit 4)

0	Full
1	Half

Parity (bits 7–5)

Bit 7	Bit 6	Bit 5	
X	X	0	Disabled
0	0	1	Odd
0	1	1	Even
1	0	1	Mark
1	1	1	Space

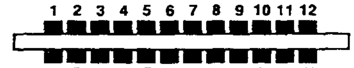
Notes

- The Command Register is optional for VIC/64/+4
- If the LF# is 128 or greater, a Line Feed will be sent after each Carriage Return
- The Secondary Address SA does not affect RS 232 operation
- Before Closing the channel, check output buffer for data with:
  VIC/64 : 100 IF PEEK(669)<>PEEK(670) THEN 100

ASCII Definitions

ACK	Acknowledge	**FS**	File Separator
BS	Backspace	**FF**	Form Feed
BEL	Bell	**GS**	Group Separator
CAN	Cancel	**HT**	Horizontal Tab
CR	Carriage Return	**LF**	Line Feed
DLE	Data Link Escape	**NAK**	Negative Ack
DEL	Delete	**NUL**	Null
DC1	Device Control 1	**RS**	Record Separator
DC2	Device Control 2	**SI**	Shift In
DC3	Device Control 3	**SO**	Shift Out
DC4	Device Control 4	**SOH**	Start Of Heading
EM	End of Medium	**STX**	Start of Text
EOT	End Of Transmission	**SUB**	Substitute
ETB	End of Xmission block	**SYN**	Synchronous Idle
ETX	End of Text	**US**	Unit Separator
ENQ	Enquiry	**VT**	Vertical Tab
ESC	Escape		

RS 232 User Port Lines

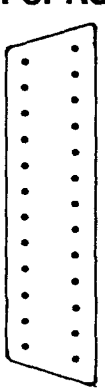


VIC 20 RS 232 is controlled by VIA 1 (6522) at $9110
C64 RS 232 is controlled by CIA 2 (6526) at $DD00
SuperPET RS 232 is controlled by ACIA (6551) at $EFF0
B Series RS 232 is controlled by ACIA (6551) at $DD00
+4 RS 232 is controlled by ACIA (6551) at $FD00

Pin#	Chip	Description	Abrv	Dir.	Modes
A	GND	Protective Ground	GND		1 2
B	FLAG2	Received Data	S_{in}	IN	1 2
C	PB0	Received Data	S_{in}	IN	1 2
D	PB1	Request to Send	RTS	OUT	1* 2
E	PB2	Data Terminal Ready	DTR	OUT	1* 2
F	PB3	Ring Indicator	RI	IN	3
H	PB4	Received line Signal	DCD	IN	2
J	PB5	Unassigned		IN	3
K	PB6	Clear To Send	CTS	IN	2
L	PB7	Data Set Ready	DSR	IN	2
M	PA2	Transmitted Data	S_{out}	OUT	1 2
N	GND	Signal Ground	GND		1 2 3

Available Modes
1) 3-Line interface (S_{in}, S_{out}, GND)
2) X-Line interface.
3) User available only (unused in code)
* these lines are held high during 3-line mode.

Pin Assignments For RS 232C Connector



Pin	Signal	Pin	Signal
1	Ground	14	Secondary Transmitted Data
2	Transmitted Data	15	Transmit Clock
3	Received Data	16	Secondary Received Data
4	Request To Send (RTS)	17	Receiver Clock
5	Clear To Send (CTS)	18	Unassigned
6	Data Set Ready (DSR)	19	Secondary Request To send
7	Logic Ground	**20**	Data Terminal Ready (DTR)
8	Carrier Detect	21	Signal Quality Detect
9	Reserved	22	Ring Detect
10	Reserved	23	Data Rate Select
11	Unassigned	24	Transmit Clock
12	Secondary Carrier Detect	25	Unassigned
13	Secondary Clear To Send		

B Series Connectors

Cartridge Connector

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

S R P N M L K J H F E D C B A

Pin	Name	Pin	Name
1	RO	A	BD0
2	A1	B	BD1
3	A2	C	BD2
4	A3	D	BD3
5	A4	E	BD4
6	A5	F	BD5
7	A6	H	BD6
8	A7	J	BD7
9	A8	K	GND
10	A9	L	GND
11	A10	M	SR/W
12	A11	N	S02
13	A12	P	CSBANK 1
14	+5 VDC	R	CSBANK 2
15	+5 VDC	S	CSBANK 3

Keyboard Connector

13 12 11 10 9 8 7 6 5 4 3 2 1

25 24 23 22 21 20 19 18 17 16 15 14

Pin	Name	Pin	Name
1	PA0	14	PA1
2	PA2	15	PA3
3	PA4	16	PA5
4	PA6	17	PA7
5	PB0	18	PC0
6	PB1	19	PC1
7	PB2	20	PC2
8	PB3	21	PC3
9	PB4	22	GND
10	PB5	23	GND
11	PB6	24	GND
12	PB7	25	PC4
13	PC5		

User Connector

1 3 5 7 9 11 13 15 17 19 21 23 25

2 4 6 8 10 12 14 16 18 20 22 24 26

Pin	Name	Pin	Name
1	GND	2	PB2
3	GND	4	PB3
5	PC	6	FLAG
7	2D7	8	2D6
9	2D5	10	2D4
11	2D3	12	2D2
13	2D1	14	2D0
15	1D7	16	1D6
17	1D5	18	1D4
19	1D3	20	1D2
21	1D1	22	1D0
23	CNT	24	+5 VDC
25	IRQ	26	SP

IEEE Connector

1 2 3 4 5 6 7 8 9 10 11 12

A B C D E F H J K L M N

(Edge-on view)

Pin	Name	Pin	Name
1	D1	A	D5
2	D2	B	D6
3	D3	C	D7
4	D4	D	D8
5	EOI	E	REN
6	DAV	F	GND
7	NRFD	H	GND
8	NDAC	J	GND
9	IFC	K	GND
10	SRQ	L	GND
11	ATN	M	GND
12	SHIELD	N	GND

Audio Jack

1 2 3

Pin	Name
1	To Speaker
2	N.C.
3	To Speaker

Power Connector

1 2 3

4 5 6

Pin	Name
1	50/60 HZ
2	-12 VDC
3	+12 VDC
4	GND
5	GND
6	+5 VDC

Video Connector

1 2 3 4 5 7

Pin	Name
1	Video
2	GND
3	Vertical Sync
4	GND
5	Horizontal Sync
6	Key
7	GND

RESET Connector

1 2

Pin	Name
1	To RESET Switch
2	To RESET Switch

Expansion Connector

60 58 56 54 52 50 48 46 44 42 40 38 36 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2

59 57 55 53 51 49 47 45 43 41 39 37 35 33 31 29 27 25 23 21 19 17 15 13 11 9 7 5 3 1

Pin	Name	Pin	Name
1	+5 VDC	2	+5 VDC
3	+5 VDC	4	+5 VDC
5	GND	6	GND
7	GND	8	GND
9	GND	10	GND
11	BRAS	12	IRQ3
13	-12 VDC	14	EXTRES
15	+12 VDC	16	S.O.
17	RES	18	LPEN
19	SR/W	20	EXTBUFCS
21	TODCLK	22	DISKROMCS
23	BOOTCLK	24	N.C.
25	S02	26	BCAS
27	S01	28	CS1
29	BD3	30	EXTPRTCS
31	BD4	32	BD2
33	BD5	34	BD1
35	DB7	36	BD0
37	BA13	38	BD7
39	BA14	40	BA15
41	BA1	42	BA0
43	BA2	44	BA11
45	BA3	46	BA10
47	BA12	48	BA4
49	BA9	50	BA5
51	BA8	52	BA6
53	BP0	54	BA7
55	BP1	56	BP2
57	NMI	58	BP3
59	RDY	60	IRQ

Co-Processor Connector

40 38 36 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4 2

39 37 35 33 31 29 27 25 23 21 19 17 15 13 11 9 7 5 3 1

Pin	Name	Pin	Name
1	EXTMA	2	DRAMO0
3	EXTMA2	4	DRAMO1
5	EXTMA7	6	DRAMO2
7	EXTMA6	8	DRAMO3
9	EXTMA5	10	DRAMO4
11	EXTMA4	12	DRAMO5
13	EXTMA1	14	DRAMO6
15	EXTMA0	16	DRAMO7
17	GND	18	GND
19	GND	20	GND
21	GND	22	BUSY 1
23	GND	24	P2REFREQ
25	GND	26	P2REFGRNT
27	GND	28	BP0
29	GND	30	BP1
31	GND	32	BP2
33	N.C.	34	BP3
35	PROCRES	36	BUSY
37	EXTBUFR/W	38	ERAS
39	DRAM R/W	40	ECAS

RS 232C Connector

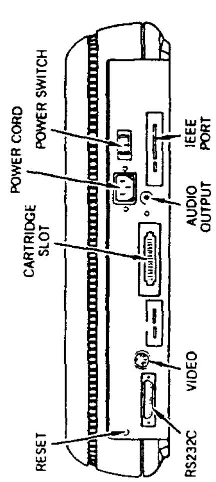
13 12 11 10 9 8 7 6 5 4 3 2 1

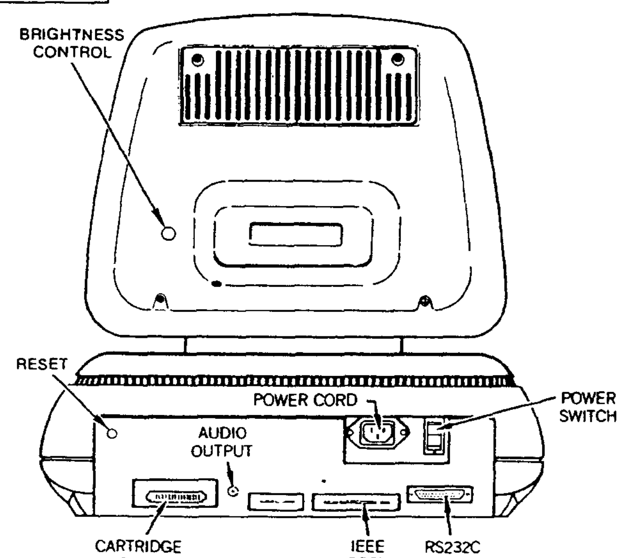
25 24 23 22 21 20 19 18 17 16 15 14

Female Connector

Pin	Name
1	SHIELD
2	TxD
3	RxD
4	RTS
5	CTS
6	DSR
7	GND
8	DCD
11	+5 VDC
18	-12 VDC
20	DTR
24	RxC

(all others N.C.)





Chip Pinouts

6502 CPU

Pin	Signal	Pin	Signal
1	V_{SS}	40	$\overline{Reset}$
2	RDY	39	Φ_2 OUT
3	Φ_1 OUT	38	S.O.
4	$\overline{IRQ}$	37	Φ_0 IN
5	N.C.	36	N.C.
6	$\overline{NMI}$	35	N.C.
7	SYNC	34	R/$\overline{W}$
8	V_{CC}	33	DB0
9	AB0	32	DB1
10	AB1	31	DB2
11	AB2	30	DB3
12	AB3	29	DB4
13	AB4	28	DB5
14	AB5	27	DB6
15	AB6	26	DB7
16	AB7	25	AB15
17	AB8	24	AB14
18	AB9	23	AB13
19	AB10	22	AB12
20	AB11	21	V_{SS}

6509 CPU

Pin	Signal	Pin	Signal
1	Ready	40	Φ_0 IN
2	$\overline{IRQ}$	39	$\overline{Reset}$
3	SYNC	38	Φ_0 OUT
4	$\overline{NMI}$	37	R/$\overline{W}$
5	AEC	36	D0
6	V_{DD}	35	D1
7	A0	34	D2
8	A1	33	D3
9	A2	32	D4
10	A3	31	D5
11	A4	30	D6
12	A5	29	D7
13	A6	28	S.O.
14	A7	27	P0
15	A8	26	P1
16	A9	25	P2
17	A10	24	P3
18	A11	23	A15
19	A12	22	A14
20	A13	21	V_{SS}

6510 CPU

Pin	Signal	Pin	Signal
1	Clk 0 IN	40	$\overline{Reset}$
2	Ready	39	Φ_2
3	$\overline{IRQ}$	38	R/$\overline{W}$
4	NMI	37	D0
5	AEC	36	D1
6	V_{CC}	35	D2
7	A0	34	D3
8	A1	33	D4
9	A2	32	D5
10	A3	31	D6
11	A4	30	D7
12	A5	29	P0
13	A6	28	P1
14	A7	27	P2
15	A8	26	P3
16	A9	25	P4
17	A10	24	P5
18	A11	23	A15
19	A12	22	A14
20	A13	21	GND

Z-80 CPU

Pin	Signal	Pin	Signal
1	A11	40	A10
2	A12	39	A9
3	A13	38	A8
4	A14	37	A7
5	A15	36	A6
6	Φ	35	A5
7	D4	34	A4
8	D3	33	A3
9	D5	32	A2
10	D6	31	A1
11	+5 V	30	A0
12	D2	29	GND
13	D7	28	$\overline{RFSH}$
14	D0	27	$\overline{M1}$
15	D1	26	$\overline{Reset}$
16	$\overline{INT}$	25	$\overline{BUSRQ}$
17	$\overline{NMI}$	24	$\overline{WAIT}$
18	$\overline{HALT}$	23	$\overline{BUSAK}$
19	$\overline{MREQ}$	22	$\overline{WR}$
20	$\overline{IORQ}$	21	$\overline{RD}$

6520 PIA
(Peripheral Interface Adapter)

Pin	Signal	Pin	Signal
1	V_{SS}	40	CA1
2	PA0	39	CA2
3	PA1	38	$\overline{IRQA}$
4	PA2	37	$\overline{IRQB}$
5	PA3	36	RS0
6	PA4	35	RS1
7	PA5	34	$\overline{Reset}$
8	PA6	33	D0
9	PA7	32	D1
10	PB0	31	D2
11	PB1	30	D3
12	PB2	29	D4
13	PB3	28	D5
14	PB4	27	D6
15	PB5	26	D7
16	PB6	25	Φ_2
17	PB7	24	CS1
18	CB1	23	$\overline{CS2}$
19	CB2	22	CS0
20	V_{CC}	21	R/$\overline{W}$

6522 VIA
(Versatile Interface Adapter)

Pin	Signal	Pin	Signal
1	V_{SS}	40	CA1
2	PA0	39	CA2
3	PA1	38	RS0
4	PA2	37	RS1
5	PA3	36	RS2
6	PA4	35	RS3
7	PA5	34	$\overline{Reset}$
8	PA6	33	D0
9	PA7	32	D1
10	PB0	31	D2
11	PB1	30	D3
12	PB2	29	D4
13	PB3	28	D5
14	PB4	27	D6
15	PB5	26	D7
16	PB6	25	Φ_2
17	PB7	24	CS1
18	CB1	23	$\overline{CS2}$
19	CB2	22	R/$\overline{W}$
20	V_{CC}	21	$\overline{IRQ}$

6526 CIA
(Complex Interface Adapter)

Pin	Signal	Pin	Signal
1	V_{SS}	40	CNT
2	PA0	39	SP
3	PA1	38	RS0
4	PA2	37	RS1
5	PA3	36	RS2
6	PA4	35	RS3
7	PA5	34	$\overline{Reset}$
8	PA6	33	DB0
9	PA7	32	DB1
10	PB0	31	DB2
11	PB1	30	DB3
12	PB2	29	DB4
13	PB3	28	DB5
14	PB4	27	DB6
15	PB5	26	DB7
16	PB6	25	Φ_2
17	PB7	24	$\overline{FLAG}$
18	$\overline{PC}$	23	$\overline{CS}$
19	TOD	22	R/$\overline{W}$
20	V_{CC}	21	$\overline{IRQ}$

6525 TPI
(Tri-Port Interface)

Pin	Signal	Pin	Signal
1	V_{SS}	40	DB7
2	PA0	39	DB6
3	PA1	38	DB5
4	PA2	37	DB4
5	PA3	36	DB3
6	PA4	35	DB2
7	PA5	34	DB1
8	PA6	33	DB0
9	PA7	32	PC7
10	PB0	31	PC6
11	PB1	30	PC5
12	PB2	29	PC4
13	PB3	28	PC3
14	PB4	27	PC2
15	PB5	26	PC1
16	PB6	25	PC0
17	PB7	24	RS0
18	$\overline{CS}$	23	RS1
19	R/$\overline{W}$	22	RS2
20	V_{DD}	21	$\overline{Reset}$

6529 SPI
(Single Port Interface)

Pin	Signal	Pin	Signal
1	R/$\overline{W}$	20	V_{DD}
2	P0	19	$\overline{CS}$
3	P1	18	DB0
4	P2	17	DB1
5	P3	16	DB2
6	P4	15	DB3
7	P5	14	DB4
8	P6	13	DB5
9	P7	12	DB6
10	V_{SS}	11	DB7

6581 - SID CHIP
(Sound Interface Device)

Pin	Signal	Pin	Signal
1	CAP1A	28	V_{DD}
2	CAP1B	27	Audio OUT
3	CAP2A	26	EXT IN
4	CAP2B	25	V_{CC}
5	$\overline{Reset}$	24	POT X
6	Φ_2	23	POT Y
7	R/$\overline{W}$	22	D7
8	$\overline{CS}$	21	D6
9	A0	20	D5
10	A1	19	D4
11	A2	18	D3
12	A3	17	D2
13	A4	16	D1
14	GND	15	D0

6551 - ACIA
(Async Communications Interface Adapter)

Pin	Signal	Pin	Signal
1	GND	28	R/$\overline{W}$
2	CS0	27	Φ_2
3	$\overline{CS1}$	26	$\overline{IRQ}$
4	$\overline{Reset}$	25	DB7
5	R_xC	24	DB6
6	XTAL1	23	DB5
7	XTAL2	22	DB4
8	$\overline{RTS}$	21	DB3
9	$\overline{CTS}$	20	DB2
10	T_xD	19	DB1
11	$\overline{DTR}$	18	DB0
12	R_xD	17	$\overline{DSR}$
13	RS0	16	$\overline{DCD}$
14	RS1	15	V_{CC}

6545-1 CRT Controller

Pin	Signal	Pin	Signal
1	GND	40	Vert Sync
2	$\overline{\text{Reset}}$	39	Hoz Sync
3	LPEN	38	RA0
4	CC0/MA0	37	RA1
5	CC1/MA1	36	RA2
6	CC2/MA2	35	RA3
7	CC3/MA3	34	RA4
8	CC4/MA4	33	DB0
9	CC5/MA5	32	DB1
10	CC6/MA6	31	DB2
11	CC7/MA7	30	DB3
12	CR0/MA8	29	DB4
13	CR1/MA9	28	DB5
14	CR2/MA10	27	DB6
15	CR3/MA11	26	DB7
16	CR4/MA12	25	$\overline{CS}$
17	CR5/MA13	24	RS
18	Display Enable	23	Φ_2
19	Cursor	22	R/$\overline{W}$
20	V_{CC}	21	CCLK

6567 VIC CHIP
(Video Interface Chip)

Pin	Signal	Pin	Signal
1	D6	40	V_{CC}
2	D5	39	D7
3	D4	38	D8
4	D3	37	D9
5	D2	36	D10
6	D1	35	D11
7	D0	34	A10
8	$\overline{IRQ}$	33	A9
9	$\overline{LP}$	32	A8
10	$\overline{CS}$	31	A7
11	R/$\overline{W}$	30	A6
12	BA	29	A5/A13
13	V_{DD}	28	A4/A12
14	Color	27	A3/A11
15	Sync + Lum	26	A2/A10
16	AEC	25	A1/A9
17	Φ_0 OUT	24	A0/A8
18	$\overline{RAS}$	23	A11
19	$\overline{CAS}$	22	Φ_1 IN
20	GND	21	Φ Color

6560/61 VIC II CHIP
(Video Interface Chip)

Pin	Signal	Pin	Signal
1	N.C.	40	V_{DD}
2	Comp Color	39	Φ_1 IN
3	Sync + Lum	38	Φ_2 IN
4	R/$\overline{W}$	37	OPTION
5	DB11	36	$P\Phi_2$
6	DB10	35	$P\Phi_1$
7	DB9	34	A13
8	DB8	33	A12
9	DB7	32	A11
10	DB6	31	A10
11	DB5	30	A9
12	DB4	29	A8
13	DB3	28	A7
14	DB2	27	A6
15	DB1	26	A5
16	DB0	25	A4
17	POT X	24	A3
18	POT Y	23	A2
19	Comp Snd	22	A1
20	V_{SS}	21	A0

2516 EPROM
2K x 8 Bits

Pin	Signal	Pin	Signal
1	A7	24	V_{CC}
2	A6	23	A8
3	A5	22	A9
4	A4	21	V_{PGM}
5	A3	20	$\overline{CS2}$
6	A2	19	A10
7	A1	18	$\overline{CS1}$
8	A0	17	D7
9	D0	16	D6
10	D1	15	D5
11	D2	14	D4
12	GND	13	D3

Low power operation when $\overline{CS}$ lines high.
V_{PGM}: Apply +25 volts to program chip memories.

2532 EPROM
4K x 8 Bits

Pin	Signal	Pin	Signal
1	A7	24	V_{CC}
2	A6	23	A8
3	A5	22	A9
4	A4	21	V_{PGM}
5	A3	20	$\overline{CS}$
6	A2	19	A10
7	A1	18	A11
8	A0	17	D7
9	D0	16	D6
10	D1	15	D5
11	D2	14	D4
12	GND	13	D3

Low power operation when $\overline{CS}$ lines high.
V_{PGM}: Apply +25 volts to program chip memories.

2564 EPROM
8K x 8 Bits

Pin	Signal	Pin	Signal
1	V_{PGM}	28	V_{CC}
2	$\overline{CS1}$	27	$\overline{CS2}$
3	A7	26	V_{CC}
4	A6	25	A8
5	A5	24	A9
6	A4	23	A12
7	A3	22	$\overline{CS}$
8	A2	21	A10
9	A1	20	A11
10	A0	19	D7
11	D0	18	D6
12	D1	17	D5
13	D2	16	D4
14	GND	15	D3

Low power operation when $\overline{CS}$ lines high.
V_{PGM}: Apply +25 volts to program chip memories.

2316 2K Static ROM
2K x 8 Bits

Pin	Signal	Pin	Signal
1	A7	24	V_{CC}
2	A6	23	A8
3	A5	22	A9
4	A4	21	$\overline{CS3}$
5	A3	20	$\overline{CS1}$
6	A2	19	A10
7	A1	18	$\overline{CS2}$
8	A0	17	D8
9	D1	16	D7
10	D2	15	D6
11	D3	14	D5
12	GND	13	D4

2332 4K Static ROM
4K x 8 Bits

Pin	Signal	Pin	Signal
1	A7	24	V_{CC}
2	A6	23	A8
3	A5	22	A9
4	A4	21	$\overline{CS2}$
5	A3	20	$\overline{CS1}$
6	A2	19	A10
7	A1	18	A11
8	A0	17	D8
9	D1	16	D7
10	D2	15	D6
11	D3	14	D5
12	GND	13	D4

2364 8K Static ROM
8K x 8 Bits

Pin	Signal	Pin	Signal
1	A7	24	V_{CC}
2	A6	23	A8
3	A5	22	A9
4	A4	21	A12
5	A3	20	$\overline{CS1}$
6	A2	19	A10
7	A1	18	A11
8	A0	17	D8
9	D1	16	D7
10	D2	15	D6
11	D3	14	D5
12	GND	13	D4

556 Dual Timer

Pin	Signal	Pin	Signal
1	Discharge 1	14	V_{CC}
2	Threshold 1	13	Discharge 2
3	Ctrl Voltage 1	12	Threshold 2
4	Reset 1	11	Ctrl Voltage 2
5	Output 1	10	Reset 2
6	Trigger 1	9	Output 2
7	GND	8	Trigger 2

4116 16K Dynamic RAM

Pin	Signal	Pin	Signal
1	−5 V	16	GND
2	Data IN	15	$\overline{CAS}$
3	R/$\overline{W}$	14	Data OUT
4	$\overline{RAS}$	13	A6
5	A5	12	A2
6	A4	11	A1
7	A3	10	A0
8	+12 V	9	+5 V

2114 Static RAM
1K x 4 Bits

Pin	Signal	Pin	Signal
1	A6	18	V_{CC}
2	A5	17	A7
3	A4	16	A8
4	A3	15	A9
5	A0	14	D0
6	A1	13	D1
7	A2	12	D2
8	$\overline{CS}$	11	D3
9	GND	10	R/$\overline{W}$

555 Timer

Pin	Signal	Pin	Signal
1	GND	8	V_{CC}
2	Trigger	7	Discharge
3	Output	6	Threshold
4	Reset	5	Ctrl Voltage

Checking Semiconductors with an Ohmmeter

P-N Diodes (including Zener, Photodiodes, or any simple P-N junction)		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Anode (forward bias)	Cathode	short or low resistance (10-1000 ohms depending on diode type)
Cathode (reverse bias)	Anode	open or high resistance (Germanium: 1M ohm typical. Silicon: 10M ohm or greater
Tunnel Diodes		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Anode (forward bias)	Cathode	short or low resistance
Cathode (reverse bias)	Anode	same, slightly lower with Cathode on +
Photoconductive Cells		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Either end	Either end	Ohmmeter reading should be equal in either direction. Resistance should increase as light decreases.
Photodiodes, LEDs, Photovoltaic Cells (LED: Short Lead = Cathode)		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Anode (forward bias)	Cathode	short or low resistance (10-1000 ohms depending on diode type)
Cathode (reverse bias)	Anode	open or high resistance (Germanium: 1Mohm typical. Silicon: 10M ohm or greater
NPN Transistors		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Emitter	Base	High resistance, unless ohmmeter voltage exceed breakdown voltage
Base	Emitter	Low resistance (forward biased junction)
Collector	Base	High resistance
Base	Collector	Low resistance, usually not as low as Emitter-Base junction since Collector is lightly doped
Emitter	Collector	High resistance, about 10-50 times less than Emitter-Base reverse bias resistance
Collector	Emitter	High resistance, slightly higher with Collector on +
PNP Transistors		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Emitter	Base	Low resistance (forward biased junction)
Base	Emitter	High resistance, unless ohmmeter voltage exceed breakdown voltage
Collector	Base	Low resistance, usually not as low as Emitter-Base junction since Collector is lightly doped
Base	Collector	High resistance
Emitter	Collector	High resistance, slightly higher with Emitter on +
Collector	Emitter	High resistance, about 10-50 times less than Base-Emitter resistance
Four-Layer Diodes, Silicon Unilateral Switches (SUS)		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Anode (forward bias)	Cathode	High resistance (1Mohm or greater)
Cathode (reverse bias)	Anode	High resistance, greater than Anode-Cathode, but immeasureable without accurate meter
DIAC, SBS		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Either end	Either end	High resistance, 1M ohm or greater
SCR (including light-activated SCR), GCS (gate-controlled switch)		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Anode (forward bias)	Cathode	High resistance, 1M ohm or greater, slightly less for hi-current SCRs
Cathode (reverse bias)	Anode	High resistance, 1M ohm or greater, usually higher than Anode-Cathode direction
Gate	Cathode	High resistance (same as P-N Diode)
Cathode	Gate	Low resistance (same as P-N Diode)
Gate	Anode	High resistance, 1M ohm or greater
Anode	Gate	High resistance, 1M ohm or greater
TRIAC		
Ohmeter + lead to	Ohmeter –lead to	Operational Results
Either Anode 1 or 2	Either Anode 2 or 1	High resistance, 1M ohm or greater, slightly less for hi-current SCRs
Gate	Anode 1	Low resistance
Anode 1	Gate	Low resistance
Gate	Anode 2	High resistance
Anode 2	Gate	High resistance

UJT (Unijunction Transistor)

Ohmeter + lead to	Ohmeter –lead to	Operational Results
Base 1	Base 2	Typically 4K-10K ohms
Base 2	Base 1	Same, 4K-10K ohms
Emitter (forward bias)	Base 1	Typically 3K-15K ohms
Base 1	Emitter	High resistance, 1M ohm or greater
Emitter (forward bias)	Base 2	Typically 2K-10K ohms, usually less than Emitter-Base 1
Base 2	Emitter	High resistance, 1M ohm or greater

Complementary UJT

Ohmeter + lead to	Ohmeter –lead to	Operational Results
Base 1	Base 2	Typically 4K-10K ohms
Base 2	Base 1	Same, 4K-10K ohms
Base 1	Emitter (forward bias)	Typically 3K-15K ohms
Emitter	Base 1	High resistance, 1M ohm or greater
Base 2	Emitter (forward bias)	Typically 2K-10K ohms, usually less than Base 1-Emitter
Emitter	Base 2	High resistance, 1M ohm or greater

Programmable UJT (PUT)

Ohmeter + lead to	Ohmeter –lead to	Operational Results
Anode	Cathode	High resistance, 1M ohm or greater
Cathode	Anode	High resistance, 1M ohm or greater
Anode	Gate	Low resistance (forward bias)
Gate	Anode	High resistance
Gate	Cathode	High resistance
Cathode	Gate	High resistance

N-Channel JFET (Field Effect Transistor)

Ohmeter + lead to	Ohmeter –lead to	Operational Results
Drain	Source	Typically 500-5K ohms
Source	Drain	Same, 500-5K ohms
Gate	Drain	Low resistance (forward biased P-N junction)
Gate	Source	Low resistance (forward biased P-N junction)
Drain	Gate	High resistance, 10M ohm or greater, unless Ohmmeter voltage exceeds JFET breakdown voltage
Source	Gate	High resistance, 10M ohm or greater, unless Ohmmeter voltage exceeds JFET breakdown voltage

P-Channel JFET

Ohmeter + lead to	Ohmeter –lead to	Operational Results
Source	Drain	Typically 500-5K ohms
Drain	Source	Same, 500-5K ohms
Drain	Gate	Low resistance (forward biased P-N junction)
Source	Gate	Low resistance (forward biased P-N junction)
Gate	Drain	High resistance, 10M ohm or greater, unless Ohmmeter voltage exceeds JFET breakdown voltage
Gate	Source	High resistance, 10M ohm or greater, unless Ohmmeter voltage exceeds JFET breakdown voltage

Enhancement MOSFET (Metal Oxide Semiconductor FET)

Ohmeter + lead to	Ohmeter –lead to	Operational Results
Drain	Source	High resistance, 10M ohm or greater
Source	Drain	High resistance, 10M ohm or greater
Gate	Drain	High resistance, 100M ohm or greater, either direction
Gate	Source	High resistance, 100M ohm or greater, either direction

Depletion MOSFET

Ohmeter + lead to	Ohmeter –lead to	Operational Results
Drain	Source	Typically 500-5K ohms
Source	Drain	Same, 500-5K ohms
Gate	Drain	High resistance, 100M ohm or greater, either direction
Gate	Source	High resistance, 100M ohm or greater, either direction

Inch Fractions

in Decimal & Millimeters

Inches			Decimal	Millimeters
1/64			0.0156	0.397
2/64	1/32		0.0313	0.794
3/64			0.0469	1.191
4/64		1/16	0.0625	1.588
5/64			0.0781	1.985
6/64	3/32		0.0938	2.381
7/64			0.1094	2.778
8/64		1/8	0.1250	3.175
9/64			0.1406	3.572
10/64	5/32		0.1563	3.969
11/64			0.1719	4.366
12/64		3/16	0.1875	4.762
13/64			0.2031	5.159
14/64	7/32		0.2188	5.556
15/64			0.2344	5.953
16/64		1/4	0.2500	6.350
17/64			0.2656	6.747
18/64	9/32		0.2813	7.144
19/64			0.2969	7.541
20/64		5/16	0.3125	7.937
21/64			0.3281	8.344
22/64	11/32		0.3438	8.731
23/64			0.3594	9.128
24/64		3/8	0.3750	9.525
25/64			0.3906	9.922
26/64	13/32		0.4063	10.319
27/64			0.4219	10.716
28/64		7/16	0.4375	11.112
29/64			0.4531	11.509
30/64	15/32		0.4688	11.906
31/64			0.4844	12.303
32/64		1/2	0.5000	12.700
33/64			0.5156	13.097
34/64	17/32		0.5313	13.494
35/64			0.5469	13.891
36/64		9/16	0.5625	14.287
37/64			0.5781	14.684
38/64	19/32		0.5938	15.081
39/64			0.6094	15.478
40/64		5/8	0.6250	15.875
41/64			0.6406	16.272
42/64	21/32		0.6563	16.669
43/64			0.6719	17.067
44/64		11/16	0.6875	17.463
45/64			0.7031	17.860
46/64	23/32		0.7188	18.238
47/64			0.7344	18.635
48/64		3/4	0.7500	19.049
49/64			0.7656	19.446
50/64	25/32		0.7813	19.842
51/64			0.7969	20.239
52/64		13/16	0.8125	20.636
53/64			0.8281	21.033
54/64	27/32		0.8438	21.430
55/64			0.8694	21.827
56/64		7/8	0.8750	22.224
57/64			0.8906	22.621
58/64	29/32		0.9063	23.018
59/64			0.9219	23.415
60/64		15/16	0.9375	23.812
61/64			0.9531	24.209
62/64	31/32		0.9688	24.606
63/64			0.9844	25.004
64/64		1.0	1.0000	25.400

International System of Units (SI)

Units Prefixes

Prefix	Symbol	Multiplier	Prefix	Symbol	Multiplier
Exa	E	10^{18}	Deci	d	10^{-1}
Peta	P	10^{15}	Centi	c	10^{-2}
Tera	T	10^{12}	Milli	m	10^{-3}
Giga	G	10^{9}	Micro	μ	10^{-6}
Mega	M	10^{6}	Nano	n	10^{-9}
Kilo	k	10^{3}	Pico	p	10^{-12}
Hecto	h	10^{2}	Femto	f	10^{-15}
Deca	da	10^{1}	Atto	a	10^{-18}

SI Base Units

Quantity	SI Unit	Symbol
Length	Meters	m
Mass	Kilograms	kg
Time	Seconds	s
Electric Current	Amperes	A
Temperature	Degrees Kelvin	K
Amount of Substance	Moles	mol
Luminous Intensity	Candela	cd

SI Supplementary Units

Quantity	SI Unit	Symbol
Plane Angle	Radians	rad
Solid Angle	Steradians	sr

SI Units Without Special Names

Quantity	SI Unit	Symbol
Area	Square Meters	m^2
Volume	Cubic Meters	m^3
Linear Velocity (Speed)	Meters/Second	m/s
Angular Velocity	Radians/Second	rad/s
Linear Acceleration	Meters/Second Squared	m/s^2
Angular Acceleration	Radians/Second Squares	rad/s^2
Wavelength	Meters	m
Density	Kilogram/Cubic Meter	kg/m^3
Concentration	Moles/Cubic Meter	mol/m^3
Specific Volume	Cubic Meters/Kilogram	m^3/kg
Luminance	Candela/Square Meter	cd/m^2
Dynamic Viscosity	Pascal Seconds	Pa × s
Kinematic Viscosity	Square Meters/Second	m^2/s
Moment of Force	Newton Meters	N × m
Surface Tension	Newton/Meter	N/m
Irradiance (Heat Flux Density)	Watts/Square Meter	W/m^2
Entropy (Heat Capacity)	Joules/Kelvin	J/K
Specific Entropy	Joules/Kilogram-Kelvin	J/(kg × K)
Specific Energy	Joules/Kilogram	J/kg
Thermal Conductivity	Watts/Meter-Kelvin	W/(m × K)
Energy Density	Joules/Cubic Meter	J/m^3
Electric Field Strength	Volts/Meter	V/m
Electric Charge Density	Coulombs/Cubic Meter	C/m^3
Surface Density of Charge (Flux Density)	Coulombs/Square Meter	C/m^2
Permittivity	Farads/Meter	F/m
Current Density	Amperes/Square Meter	A/m^2
Magnetic Field Strength	Amperes/Meter	A/m
Permeability	Henries/Meter	H/m
Molar Energy	Joules/Mole	J/mol
Molar Entropy	Joules/Mole Kelvin	J/(mol × K)
Radiant Intensity	Watts/Steradian	W/sr
Radiance	Watts/Square Meter Steradian	$W/(m^2 \times sr)$
Exposure	Coulombs/Kilogram	C/kg
Absorbed Dose Rate	Grays/Second	Gy/s

SI Units With Special Names

Quantity	SI Unit	Symbol	Derivative
Frequency	Hertz	Hz	1/s or s^{-1}
Force	Newtons	N	$m \times kg/s^2$
Pressure, Stress	Pascals	Pa	N/m^2
Energy, Work, Quantity of Heat	Joules	J	N × m
Quantity of Heat	Calories	cal	
Power, Radiant Flux	Watt	W	J/s
Quantity of Electricity, Electric Charge	Coulombs	C	s × A
Electric Potential, Potential Difference			
Electromotive Force	Volts	V	W/A
Electric Capacitance	Farads	F	C/V
Electric Resistance	Ohms	ψ	V/A
Electric Conductance	Siemens	S	A/V
Magnetic Flux	Webers	Wb	V × s
Magnetic Flux Density	Tesla	T	Wb/m^2
Inductance	Henries	H	Wb/A
Luminous Flux	Lumens	lm	cd × sr.
Illuminance	Lux	lx	lm/m^2
Activity of Radionuclides	Becquerals	Bq	s^{-1}
Absorbed Dose of Ionising Radiation	Grays	Gy	J/kg

Names For Large Numbers

Name	French & US. Equivalent	Number of Zeros	British & German Equivalent	Number of Zeros
million	1000 thousands	6	1000 thousands	6
milliard	1000 millions	9	1000 millions	9
billion	1000 millions	9	1,000,000 millions	12
trillion	1000 billions or	12	1,000,000 billions or	18
	1,000,000 millions	12	1,000,000 million millions	18
quadrillion	1000 trillions	15	1,000,000 trillions	24
quintillion	1000 quadrillions	18	1,000,000 quadrillions	30
sextillion	1000 quintillions	21	1,000,000 quintillions	36
septillion	1000 sextillions	24	1,000,000 sextillions	42
octillion	1000 septillions	27	1,000,000 septillions	48
nonillion	1000 octillions	30	1,000,000 octillions	54
decillion	1000 nonillions	33	1,000,000 nonillions	60
undecillion	1000 decillions	36	1,000,000 decillions	66
duodecillion	1000 undecillions	39	1,000,000 undecillions	72
tredecillion	1000 duodecillions	42	1,000,000 duodecillions	78
quattuordecillion	1000 tredecillions	45	1,000,000 tredecillions	84
quindecillion	1000 quattuordecillions	48	1,000,000 quattuordecillions	90
sexdecillion	1000 quindecillions	51	1,000,000 quindecillions	96
septendecillion	1000 sexdecillions	54	1,000,000 sexdecillions	102
octodecillion	1000 septendecillions	57	1,000,000 septendecillions	108
novemdecillion	1000 octodecillions	60	1,000,000 octodecillions	114
vigintillion	1000 novemdecillions	63	1,000,000 novemdecillions	120

Constant Values

Constant	Symbol	Value
Absolute Zero		−273.15°C or −459.7°F
Ampere's Circuital Law Constant	K	2×10^{-7} Newtons/Amp^2
Avogadro's Number	N_o	6.022169×10^{23}
Bohr Magneton	μ_B	9.274096×10^{-24}Joules/Second
Boltzmann's Constant	k	1.380622×10^{-23}Joules/Degrees Kelvin
Coulomb's Law Constant	k	$8.988 \times$ m10^9 Newton Meters Squared/Coulomb^2
Electron Charge	e	$1.6021917 \times 10^{-19}$C
Electron Charge To Mass Ratio	e/m_c	1.7588028×10^{11}C/Kilogram
Faraday Constant	F	9.648670×10^7 C k mole^{-1}
Gas Constant	R_o	8.31434×10^3J-k mole^{-1}K^{-1}
Gravitational Constant	G	6.6732×10^{-11}Cubic Meters/Kilogram Seconds^2
Planck's Constant	h	6.626196×10^{-34} Joule-Seconds
Rydberg Constant	R_{∞}	1.09737312×10^7m^{-1}
Speed of Light	C	2.9979250×10^8 Meters/Second
Speed of Sound (in air at 28° C)		746 Miles/Hour
Speed of Sound (in air at 28° C)		348 Meters/Second
Earth Orbiting Satellite		7.5 Kilometers/Second (approx.)
Earth Orbiting Satellite		17000 Miles/Hour (approx.)
Compton Electron Wavelength	λ_c	$2.4263096 \times 10^{-12}$ Meters
Compton Proton Wavelength	λ_{cp}	$1.3214409 \times 10^{-15}$ Meters
Compton Neutron Wavelength	λ_{cn}	$1.3196217 \times 10^{-15}$ Meters
Electron Magnetic Moment	μ_c	9.284851×10^{-24}Joules/Second
Proton Magnetic Moment	μ_p	$1.4106203 \times 10^{-28}$Joules/Second
Electron Rest Mass	m_e	9.109558×10^{-31} Kilograms
	m_e	5.485930×10^{-4}Atomic Mass Units
Proton Rest Mass	M_p	1.672614×10^{-27} Kilograms
	M_p	1.00727661 Atomic Mass Units
Neutron Rest Mass	M_n	1.674920×10^{-27} Kilograms
	M_n	1.00866520 Atomic Mass Units

Mathematical Functions

Function	BASIC Equivalent
Secant	SEC(X) = 1 / COS(X)
Cosecant	CSC(X) = 1 / SIN(X)
Cotangent	COT(X) = 1 / TAN(X)
Inverse Sine	ARCSIN(X) = ATN(X / SQR(−X*X + 1))
Inverse Cosine	ARCCOS(X) = ATN(X / SQR(−X*X + 1)) + π/2
Inverse Secant	ARCSEC(X) = ATN(X / SQR(X*X−1))
Inverse Cosecant	ARCCSC(X) = ATN(X / SQR(X*X−1) + (SGN(X) − 1*π/2)
Inverse Cotangent	ARCCOT(X) = ATN(X) + π/2
Hyperbolic Sine	SINH(X) = (EXP(X) − EXP(−X)) / 2
Hyperbolic Cosine	COSH(X) = (EXP(X) + EXP(−X)) / 2
Hyperbolic Tangent	TANH(X) = EXP(−X) / (EXP(X) + EXP(−X)) * 2 + 1
Hyperbolic Secant	SECH(X) = 2 / (EXP(X) + EXP(−X)
Hyperbolic Cosecant	CSCH(X) = 2 / (EXP(X) − EXP(−X)
Hyperbolic Cotangent	COTH(X) = EXP(−X) / (EXP(X) − EXP(−X)) * 2 + 1
Inverse Hyperbolic Sine	ARCSINH(X) = LOG(X + SQR(X*X + 1))
Inverse Hyperbolic Cosine	ARCCOSH(X) = LOG(X / SQR(X*X−1))
Inverse Hyperbolic Tangent	ARCTANH(X) = LOG((1 + X) / (1−X)) / 2
Inverse Hyperbolic Secant	ARCSECH(X) = LOG(SQR(−X*X + 1) + 1/X)
Inverse Hyperbolic Cosecant	ARCCSCH(X) = LOG(X / SQR(X*X−1) + (SGN(X) − 1*π/2)
Inverse Hyperbolic Cotangent	ARCCOTH(X) = LOG(X) + π/2

Roman Numerals

I	1	XI	11	XXX	30	CD	400
II	2	XII	12	XL	40	D	500
III	3	XIII	13	L	50	DC	600
IV	4	XIV	14	LX	60	DCC	700
V	5	XV	15	LXX	70	DCCC	800
VI	6	XVI	16	LXXX	80	CM	900
VII	7	XVII	17	XC	90	M	1000
VIII	8	XVIII	18	C	100	MCM	1900
IX	9	XIX	19	CC	200	MM	2000
X	10	XX	20	CCC	300	$\overline{V}$	5000

Rules:

1. An overhead line indicates the value multiplied by 1000.
2. Repeating a letter repeats its value (XX = 20, CCC = 300)

Boolean Truth Table

AND	OR	NOT	XOR
1 AND 1 = 1 1 AND 0 = 0 0 AND 1 = 0 0 AND 0 = 0	1 OR 1 = 1 1 OR 0 = 1 0 OR 1 = 1 0 OR 0 = 0	NOT 0 = 1 NOT 1 = 0	1 XOR 1 = 0 1 XOR 0 = 1 0 XOR 1 = 1 0 XOR 0 = 0
Result is 1 if both bits are 1	Result is 1 if either bit is 1	Each bit is complemented	Result is 1 if one or the other but not both

Force Formulae

Force = Mass × Acceleration

Horsepower

1 HP = 33000 Foot-Pounds of Work per Minute

Torque

Torque = Force × Radius
Torque = 63025 × Horsepower / RPM

Centrifugal Force

Centrifugal Force (outward) = Centripetal Force (inward)
Centrifugal Force = Weight × Linear Velocity^2 / (32.16 × Radius)
Centrifugal Force = Weight × Radius × RPM^2 / 2932.55
Centrifugal Force = 1.22760 × Weight × Radius × RPS^2

Weight is in pounds — RPM is in revolutions/minute
Linear Velocity is in feet/second — RPS is in revolutions/second
Radius is in feet

Propeller Thrust

Typical Thrust for a power boat:

Prop Thrust = 33000 × Motor Horsepower × Prop Efficiency / Speed
Prop Thrust = 33000 × Motor HP × Prop Effcy / (Prop Pitch × RPMs)
Where Prop Efficiency in water ranges from 60% to 70% (65% practically)
Speed is in feet/minute
Prop Pitch is in feet
RPMs is RPMs @ n Motor Horsepower

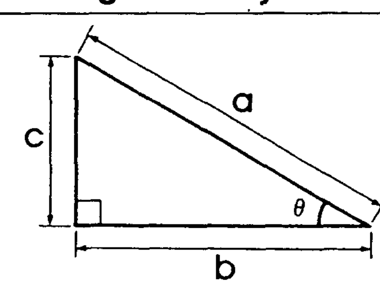
Typical Thrust for an airplane in level flight:

Prop Thrust = 375 × Motor Horsepower × Prop Efficiency / MPH
Where Prop Efficiency in air ranges from 70% to 87% (80% practically)

Gravity

X = Forward Velocity × Time
Y = Upward Velocity × Time − ½ Gravity × Time^2
Where Gravity on Earth at Sea Level is 32.2 Feet/Second^2

Trigonometry Rules



SIN θ	c / a	Opposite / Hypotenuse
COS θ	b / a	Adjacent / Hypotenuse
TAN θ	c / b	Opposite / Adjacent
CSC θ	a / c	Hypotenuse / Opposite
SEC θ	a / b	Hypotenuse / Adjacent
COT θ	b / c	Adjacent / Hypotenuse

Unit Conversion Table

Avoirdupois: indicates regular English measure – based on 16 ounces to the pound.

To Convert:	Multiply by:	To Get:
	A	
Abcoulombs	2.998×10^{10}	Statcoulombs
Acres	160	Rods
Acres	10	Square Chains (Gunters)
Acres	43560	Square Feet
Acres	0.4047	Hectares
Acres	100000	Square Links (Gunters)
Acres	4047	Square Meters
Acres	0.0016	Square Miles
Acres	4840	Square Yards
Acre Feet	43560	Cubic Feet
Acre Feet	1233.48	Cubic Meters
Acre Feet	3.259×10^{5}	Gallons
Amperes/Square Centimeters	6.452	Amps/Square Inch
Amperes/Square Inch	0.1550	Amps/Square Centimeter
Ampere-Hours	3600	Coulombs
Ampere-Hours	0.03731	Faradays
Ampere-Turns	1.257	Gilberts
Ampere-Turns/Inch	0.4950	Gilberts/Centimeter
Ampere-Turns/Meter	0.01257	Gilberts/Centimeter
Angstroms	3937×10^{-9}	Inches
Angstroms	10^{-10}	Meters
Angstroms	10^{-4}	Microns
Ares	0.02471	Acres (US.)
Ares	119.60	Square Yards
Ares	100	Square Meters
Arpents (French measure)	58.47131	Meters
Arpents (French area measure)	0.3418894	Hectares
Astronomical Units	1.49597870×10^{8}	Kilometers
Atmospheres (atm.)	76.0	Centimeters–Mercury
Atmospheres	33.90	Feet of Water (at 4° C)
Atmospheres	29.92	Inches–Mercury (at 0° C)
Atmospheres	1.0333	Kilogram/Square Centimeters
Atmospheres	14.70	Pounds/Square Inch
Atmospheres	1.058	Tons/Square Foot
Atmospheres	0.007348	Tons/Square Inch
Atomic Mass Units (amu)	1.660531×10^{-27}	Kilograms
	B	
Barrels (US.) (dry)	7056	Cubic Inches
Barrels (US.) (dry)	105	Quarts (dry)
Barrels (US.) (liquid)	31.5	Gallons (US.)
Barrels (oil)	42	Gallons (oil)
Bars	0.9869	Atmospheres
Bars	10^{6}	Dynes/Square Centimeter
Bars	1.020×10^{4}	Kilograms/Square Meter
Bars	2089	Pounds/Square Foot
Bars	14.50	Pounds/Square Inch
Baryls	1.0	Dynes/Square Centimeter
Bolts (US.) (cloth)	36.576	Meters
Board Feet	2359.7	Cubic Centimeters
Board Feet	144	Cubic Inches
British Thermal Units (BTU)	1.0550×10^{10}	Ergs
BTU	778.3	Foot-Pounds
BTU	252.0	Gram-Calories
BTU	3.931×10^{-4}	Horsepower-Hours
BTU	1054.8	Joules
BTU	2.928×10^{-4}	Kilowatt-Hours
BTU	107.5	Kilowatt-Meters
BTU	10.409	Liter-Atmospheres
BTU/Hour	0.2162	Foot-Pounds/Second
BTU/Hour	0.0700	Gram-Calories/Second
BTU/Hour	3.929×10^{-4}	Horsepower-Hours
BTU/Hour	0.2931	Watts
BTU/Minute	12.96	Foot-Pounds/Second
BTU/Minute	0.02356	Horsepower
BTU (thermochemical)/Minute	17.57250	Watts
BTU (International)/Minute	17.58426	Watts
BTU/Square Foot/Minute	0.1221	Watts/Square Inch
Bucket (British) (dry)	1.818×10^{4}	Cubic Centimeters
Bushel (struck measure)	4	Pecks
Bushel (struck measure)	32	Dry Quarts
Bushel (struck measure)	1.2445	Cubic Feet
Bushel (struck measure)	2150.42	Cubic Inches
Bushel (struck measure)	35.238	Liters
Bushel (struck measure)	64.0	Pints (dry)
Bushel (struck measure)	32.0	Quarts (dry)
Bushel (heaped)	1.278	Bushels (struck measure)
Bushel (heaped)	2747.715	Cubic Inches
	C	
Calory-grams	3.96832×10^{-3}	British Thermal Units
Candle/Square Centimeter	3.142	Lamberts
Candle/Square Inch	0.4870	Lamberts
Carat (c.)	3.086	Grains
Carat	200	Milligrams
Celsius	(C x 9/5) + 32	Fahrenheit
Centares	1.0	Square Meters
Centigrams (cgm.)	0.01	Grams
Centiliters (cl.)	0.3382	Ounces (US. liquid)

To Convert:	Multiply by:	To Get:
Centiliters	0.6103	Cubic Inches
Centiliters	2.705	Drams
Centimeters (cm.)	0.3937	Inches
Centimeters	10	Millimeters
Centimeters	393.7	Mils
Centimeters	0.01094	Yards
Centimeters/Second	1.1969	Feet/Minute
Centimeters/Second	0.03281	Feet/Second
Centimeters/Second	0.036	Kilometers/Hour
Centimeters/Second	0.1943	Knots
Centimeters/Second	0.6	Meters/Minute
Centimeters/Second	0.02237	Miles/Hour
Centimeters/Second	3.728×10^{-4}	Miles/Minute
Centimeter-Dynes	1.020×10^{-3}	Centimeter-Grams
Centimeter-Dynes	1.020×10^{-8}	Meter-Kilograms
Centimeter-Dynes	7.376×10^{-8}	Pound-Feet
Centimeter-Grams	980.7	Centimeter-Dynes
Centimeter-Grams	10^{-5}	Meter-Kilograms
Centimeter-Grams	7.233×10^{-5}	Pound-Feet
Centimeters of Mercury	0.01316	Atmospheres
Centimeters of Mercury	0.4461	Feet of Water
Centimeters of Mercury	136.0	Kilograms/Square Meter
Centimeters of Mercury	27.85	Pounds/Square Foot
Centimeters of Mercury	0.1934	Pounds/Square Inch
Central	100	Pounds
Central	45.359	Kilograms
Chains	66.0	Feet
Chains	792.0	Inches
Chains	20.1168	Meters
Chains	22.00	Yards
Circular Mils	5.067×10^{-6}	Square Centimeters
Circular Mils	7.854×10^{-7}	Square Inches
Circular Mils	0.7854	Square Mils
Circumference	6.283	Radians
Coal Tubs (NFLD.)	100.0	Pounds
Cord (stacked wood)	3.6246	Cubic Meters
Cord (stacked wood)	128	Cubic Feet
Coulombs	2.998×10^{9}	Statcoulombs
Coulombs	6.242×10^{18}	Elem. Ch.
Coulombs	1.036×10^{-5}	Faradays
Coulombs/Square Centimeter	64.52	Coulombs/Square Inch
Cubic Centimeters (cc.)	3.531×10^{-5}	Cubic Feet
Cubic Centimeters	0.061023	Cubic Inches
Cubic Centimeters	1×10^{-6}	Cubic Meters
Cubic Centimeters	1.3079×10^{-6}	Cubic Yards
Cubic Centimeters	2.642×10^{-4}	Gallons (US.)
Cubic Centimeters	2.199×10^{-4}	Gallons (Imp.)
Cubic Centimeters	0.0010	Liters
Cubic Centimeters	1.0	Milliliters
Cubic Centimeters	0.0021	Pints (liquid)
Cubic Centimeters	0.0011	Quarts (liquid)
Cubic Feet	1728	Cubic Inches
Cubic Feet	0.02831685	Cubic Meters
Cubic Feet	7.48052	Gallons (US. liquid)
Cubic Feet	28.317	Liters
Cubic Feet	59.84	Pints (US. liquid)
Cubic Feet	29.92	Quarts (US. liquid)
Cubic Feet/Minute	472.0	Cubic Centimeters/Second
Cubic Feet/Minute	0.1247	Gallons/Second
Cubic Feet/Minute	0.4719	Liters/Second
Cubic Feet/Minute	0.0011	Quarts (liquid)
Cubic Feet/Minute	0.0011	Quarts (liquid)
Cubic Feet/Second	448.831	Gallons/Minute
Cubic Feet/Second	0.646317	Million Gallons/Day
Cubic Feet Aluminum	169	Pounds of Aluminum
Cubic Feet Brass	520	Pounds of Brass
Cubic Feet Brick	125 (approx.)	Pounds of Brick
Cubic Feet Cast Iron	450	Pounds of Cast Iron
Cubic Feet Concrete	145	Pounds of Concrete
Cubic Feet Copper	555	Pounds of Copper
Cubic Feet Cork	15	Pounds of Cork
Cubic Feet Glass	160-180	Pounds of Glass
Cubic Feet Gold	1204	Pounds of Gold
Cubic Feet Hardwood	45 (approx.)	Pounds of Hardwood
Cubic Feet Ice	57	Pounds of Ice
Cubic Feet Lead	708	Pounds of Lead
Cubic Feet Silver	655	Pounds of Silver
Cubic Feet Softwood	30 (approx.)	Pounds of Softwood
Cubic Feet Steel	490	Pounds of Steel
Cubic Feet Water	62.43	Pounds of Water
Cubic Inches	16.387	Cubic Centimeters
Cubic Inches	0.0005787	Cubic Feet
Cubic Inches	1.6387×10^{-5}	Cubic meters
Cubic Inches	2.1433×10^{-5}	Cubic Yards
Cubic Inches	0.004329	Gallons (US.)
Cubic Inches	0.003605	Gallons (Imp.)
Cubic Inches	0.016387	Liters
Cubic Inches	1.061×10^{5}	Mil-Feet
Cubic Inches	4.433	Drams (liquid)

To Convert:	Multiply by:	To Get:
Cubic Inches	0.554	Ounces (liquid)
Cubic Inches	0.03463	Pints (US. liquid)
Cubic Inches	0.01732	Quarts (US. liquid)
Cubic Meters	1×10^{6}	Cubic Centimeters
Cubic Meters	35.31	Cubic Feet
Cubic Meters	61023	Cubic Inches
Cubic Meters	1.308	Cubic Yards
Cubic Meters	264.2	Gallons (US.)
Cubic Meters	220.0	Gallons (Imp.)
Cubic Meters	1000	Liters
Cubic Meters	2113	Pints (US. liquid)
Cubic Meters	1759.4	Pints (Imp. liquid)
Cubic Meters	1057	Quarts (US. liquid)
Cubic Meters	880.1	Quarts (Imp. liquid)
Cubic Tons	40	Cubic Feet
Cubic Tons	1.1327	Cubic Meters
Cubic Yards	27	Cubic Feet
Cubic Yards	46.656	Cubic Inches
Cubic Yards	0.76456	Cubic Meters
Cubic Yards	202.0	Gallons (US.)
Cubic Yards	168.2	Gallons (Imp.)
Cubic Yards	764.5	Liters
Cubic Yards	1615.9	Pints (US. liquid)
Cubic Yards	807.9	Quarts (US. liquid)
Cubic Yards	1345.5	Pints (Imp. liquid)
Cubic Yards	672.7	Quarts (Imp. liquid)
Cubic Yards/Minute	0.45	Cubic Feet/Second
Cubic Yards/Minute	3.367	Gallons/Second
Cubic Yards/Minute	12.74	Liters/Second
Cunits (timber)	100.0	Cubic Feet
Cunits (timber)	2.83168	Cubic Meters
Cups (Cdn.)	227.0	Milliliters
Cups (US.)	236.0	Milliliters
Cups (measuring)	8	Ounces (liquid)
Cups (measuring)	0.5	Pints (liquid)
Cups (measuring)	16	Tablespoons
	D	
Dalton	1.650×10^{-24}	Grams
Days	86400	Seconds
Degrees (angle)	1.1111	Grads
Degrees (angle)	60	Minutes
Degrees (angle)	0.01111	Quadrants
Degrees (angle)	0.01745 (or $\pi/180$)	Radians
Degrees (angle)	3600	Seconds
Degrees/Second	0.01745	Radians/Second
Degrees/Second	0.1667	Revolutions/Minute
Degrees/Second	0.002778	Revolutions/Second
Dekaliter (dkl.)	2.642	Gallons (US.)
Dekaliter (dkl.)	3.1729	Gallons (Imp.)
Dekaliter (dkl.)	1.135	Pecks
Drams (dr.) (avoirdupois)	27.3437	Grains
Drams (dr. ap.) (apothecaries')	60	Grains
Drams (apothecaries')	3.888	Grams
Drams (apothecaries')	0.1371429	Ounces (avoirdupois)
Drams (apothecaries')	0.125	Ounces (apothecaries')
Drams (fl. dr.) (liquid) (avoirdupois)	0.0625	Ounces
Drams (liquid) (avoirdupois)	0.2256	Cubic Inches
Drams (liquid) (avoirdupois)	3.6967	Milliliters
Drams (avoirdupois)	1.7718	Grams
Drams (liquid) (British)	0.217	Cubic Inches
Drams (liquid) (British)	0.961	Drams (US. liquid)
Drams (liquid) (British)	3.552	Milliliters
Drops (Cdn. Hospital)	0.01	Teaspoons
Drops (Cdn. Hospital)	0.05	Milliliters
Dynes	1.020×10^{-3}	Grams
Dynes	10^{-7}	Joules/Centimeter
Dynes	10^{-5}	Joules/Meter (Newtons)
Dynes	7.233×10^{-5}	Poundals
Dynes	2.248×10^{-6}	Pounds
Dynes/Centimeter	0.01	Ergs/Square Millimeter
Dynes/Square Centimeter	10^{-6}	Bars
Dynes/Square Centimeter	9.869×10^{-7}	Atmospheres
Dynes/Square Centimeter	2.953×10^{-5}	Inches of Mercury (at 0° C)
Dynes/Square Centimeter	4.015×10^{-4}	Inches of Water (at 4° C)
	E	
Ells	114.30	Centimeters
Ells	45.0	Inches
Ergs	9.480×10^{-11}	BTU
Ergs	1.0	Dyne-Centimeters
Ergs	7.3756103×10^{-8}	Foot-Pounds
Ergs	0.2389×10^{-7}	Gram-Calories
Ergs	1.020×10^{-3}	Gram-Centimeters
Ergs	3.7250×10^{-14}	Horsepower-Hours
Ergs	10^{-7}	Joules
Ergs	0.2778×10^{-13}	Kilowatt-Hours
Ergs/Second	5.688×10^{-6}	BTU/Minute
Ergs/Second	4.427×10^{-6}	Foot-Pounds/Minute
Ergs/Second	7.3756×10^{-8}	Foot-Pounds/Second
Ergs/Second	1.341×10^{-10}	Horsepower
Ergs/Second	1.433×10^{-9}	Kilogram-Calories/Minute
Ergs/Second	10^{-10}	Kilowatts
	F	
Farads	10^{6}	Microfarads

To Convert:	Multiply by:	To Get:
Faradays	26.80	Ampere-Hours
Faradays	9.649×10^{4}	Coulombs
Faradays/Second	9.649×10^{4}	Amperes (absolute)
Fahrenheit	$(F - 32) \times 5/9$	Celsius
Fathoms	6	Feet
Fathoms	1.828804	Meters
Feet	0.3048	Meters
Feet (French measure)	0.324841	Meters
Feet (US. survey, limited use)	0.3048006	Meters
Feet	1.2×10^{-4}	Mils
Feet	1.645×10^{-4}	Nautical Miles
Feet	1.894×10^{-4}	Statute Miles
Feet of Water	0.02950	Atmospheres
Feet of Water	0.8826	Inches of Mercury
Feet of Water	0.03048	Kilograms/Square Centimeter
Feet of Water	62.43	Pounds/Square Feet
Feet of Water	0.4335	Pounds/Square Inch
Feet/Minute	0.5080	Centimeters/Second
Feet/Minute	0.01829	Kilometers/Hour
Feet/Minute	0.3048	Meters/Minute
Feet/Minute	0.01136	Miles/Hour
Feet/Second	30.48	Centimeters/Second
Feet/Second	1.097	Kilometers/Hour
Feet/Second	0.5921	Knots
Feet/Second	18.29	Meters/Minute
Feet/Second	0.6818	Miles/Hour
Feet/Second	0.01136	Miles/Minute
Firkins	9.0	Gallons
Firkins	40.91	Liters
Foot-Pounds	1.286×10^{-3}	British Thermal Units (BTU)
Foot-Pounds	1.356×10^{7}	Ergs
Foot-Pounds	0.3238	Gram-Calories
Foot-Pounds	5.0505×10^{-7}	Horsepower-Hours
Foot-Pounds	1.356	Joules
Foot-Pounds	0.1383	Kilogram-Meters
Foot-Pounds	3.766×10^{-7}	Kilowatt-Hours
Foot-Pounds/Minute	0.01667	Foot-Pounds/Second
Foot-Pounds/Minute	3.030×10^{-5}	Horsepower
Foot-Pounds/Minute	2.2597×10^{-5}	Kilowatts
Foot-Pounds/Second	4.6263	BTU/Hour
Foot-Pounds/Second	0.07717	BTU/Minute
Foot-Pounds/Second	1.818×10^{-3}	Horsepower
Foot-Pounds/Second	0.01945	Kilogram-Calories/Minute
Foot-Pounds/Second	1.356×10^{-3}	Kilowatts
Furlongs	660	Feet
Furlongs	201.168	Meters
Furlongs	0.125	Miles
Furlongs	40	Rods
Furlongs	220	Yards
	G	
Gallons (gal.)	8	Pints (liquid)
Gallons	4	Quarts (liquid)
Gallons Imperial	1.2009	U.S. Gallons
Gallons U.S.	0.8327	Imperial Gallons
Gallons (US.)	3785	Cubic Centimeters
Gallons (US.)	0.1337	Cubic Feet
Gallons (US.)	231	Cubic Inches
Gallons (US.)	0.0038	Cubic Meters
Gallons (US.)	1024	Drams (liquid)
Gallons (US.)	3.785	Liters
Gallons (US.)	32	Gills (liquid)
Gallons (US.)	128	Ounces (US. liquid)
Gallons (Imp.)	4545.6	Cubic Centimeters
Gallons (Imp.)	0.1606	Cubic Feet
Gallons (Imp.)	277.42	Cubic Inches
Gallons (Imp.)	0.00456	Cubic Meters
Gallons (Imp.)	1229.77	Drams (liquid)
Gallons (Imp.)	4.5456	Liters
Gallons (Imp.)	38.43	Gills (liquid)
Gallons (Imp.)	160	Ounces (Imp. liquid)
Gallons (US.) of Water	6.9489	Pounds of Water
Gallons (Imp.) of Water	8.3453	Pounds of Water
Gausses	6.452	Lines/Square Inch
Gausses	10^{-8}	Webers/Square Centimeter
Gausses	6.452×10^{-8}	Webers/Square Inch
Gilberts	0.7958	Ampere-Turns
Gilberts/Centimeter	2.021	Ampere-Turns/Inch
Gilberts/Centimeter	79.58	Ampere-Turns/Meter
Gill (gi.)	142.07	Cubic Centimeters
Gill	7.219	Cubic Inches
Gill	4	Ounces (US. liquid)
Gill	0.118	Liters
Grade	0.01571	Radians
Grads	0.90	Degrees (angle)
Grains (troy or apothecaries')	1.0	Grains (avoirdupois)
Grains	64.799	Milligrams
Grains	2.286×10^{-3}	Ounces (avoirdupois)
Grains	0.04167	Pennyweight (troy)
Grains/US. Gallon	17.118	Parts/Million
Grains/Imp. Gallon	14.286	Parts/Million
Grains/US. Gallon	142.86	Pounds/Million Gallons
Grams (g.)	980.7	Dynes
Grams	15.432	Grains
Grams	9.807×10^{-5}	Joules/Centimeter

To Convert:	Multiply by:	To Get:
Grams	9.807×10^{-3}	Newtons
Grams	0.03527	Ounces (avoirdupois)
Grams	0.03215	Ounces (troy)
Grams	0.07093	Poundals
Grams	0.002205	Pounds
Gram-Calories	3.9683×10^{-3}	BTU
Gram-Calories	4.1868×10^{7}	Ergs
Gram-Calories	3.0880	Foot-Pounds
Gram-Calories	1.5596×10^{-6}	Horsepower-Hours
Gram-Calories	1.1630×10^{-6}	Kilowatt-Hours
Gram-Calories/Second	14.286	BTU/Hour
Gram-Centimeters	9.297×10^{-8}	BTU
Gram-Centimeters	980.7	Ergs
Gram-Centimeters	9.807×10^{-5}	Joules
Gram-Centimeters	2.343×10^{-8}	Kilogram-Calories
Gram-Centimeters	10^{-5}	Kilogram-Meters
Grams/Centimeter	5.6×10^{-3}	Pounds/Inch
Grams/Cubic Centimeter	62.43	Pounds/Cubic Feet
Grams/Cubic Centimeter	0.03613	Pounds/Cubic Inch
Grams/Cubic Centimeter	3.405×10^{-7}	Pounds/Mil-Foot
Grams/Liter	58.417	Grains/Gallon (US.)
Grams/Liter	1000.0	Parts/Million
Grams/Liter	8.345	Pounds/1000 Gallons
Grams/Liter	0.062427	Pounds/Cubic Feet
Grams/Square Centimeter	2.0481	Pounds/Square Feet

H

To Convert:	Multiply by:	To Get:
Hand	10.16	Centimeters
Hectares	2.471	Acres
Hectares	1.076×10^{5}	Square Feet
Hectoliter (hl.)	26.418	Gallons
Hectoliter	2.838	Bushels
Hogsheads (British)	10.114	Cubic Feet
Hogsheads (US.)	8.42184	Cubic Feet
Hogsheads (US.)	63.0	Gallons (US.)
Hogsheads (US.)	52.4	Gallons (Imp.)
Hogsheads (US.)	236.4	Liters
Horsepower	1.014	Horsepower metric
Horsepower (metric)	0.9863	Horsepower
Horsepower	42.44	BTU/Minute
Horsepower	33000	Foot-Pounds/Minute
Horsepower	550	Foot-Pounds/Second
Horsepower (metric)	542.5	Foot-Pounds/Second
Horsepower	10.68	Kilogram-Calories/Minute
Horsepower	0.7457	Kilowatts
Horsepower (boiler)	33479	BTU/Hour
Horsepower (boiler)	9.803	Kilowatts
Horsepower Hours	2547	BTU
Horsepower Hours	2.6845×10^{13}	Ergs
Horsepower Hours	1.98×10^{6}	Foot-Pounds
Horsepower Hours	641190	Gram-Calories
Horsepower Hours	2.6845×10^{6}	Joules
Horsepower Hours	2.737×10^{5}	Kilogram-Meters
Hours	0.04167	Days
Hours	0.005952	Weeks
Hundredweights (cwt.) (gross or long)	112	Pounds
Hundredweights (gross or long)	50.802	Kilograms
Hundredweights (gross or long)	0.05	Tons (long)
Hundredweights (net cwt)(net or short)	1600	Ounces (avoirdupois)
Hundredweights (net or short)	100	Pounds
Hundredweights (net or short)	45.359	Kilograms
Hundredweights (net or short)	0.0453592	Tons (metric)
Hundredweights (net or short)	0.0446429	Tons (long or gross)

I

To Convert:	Multiply by:	To Get:
Inches	2.540	Centimeters
Inches	1.578×10^{-5}	Miles
Inches	1000	Mils
Inches	6	Picas (typography)
Inches	72	Points (typography)
Inches	2.778×10^{-2}	Yards
Inches of Mercury	0.03342	Atmospheres
Inches of Mercury	1.133	Feet of Water
Inches of Mercury	0.03453	Kilograms/Square Centimeter
Inches of Mercury	70.73	Pounds/Square Foot
Inches of Mercury	0.4912	Pounds/Square Inch
Inches of Water (at 4° C)	2.458×10^{-3}	Atmospheres
Inches of Water (at 4° C)	0.07355	Inches of Mercury
Inches of Water (at 4° C)	2.540×10^{-3}	Kilograms/Square Centimeter
Inches of Water (at 4° C)	0.5781	Ounces/Square Inch
Inches of Water (at 4° C)	5.204	Pounds/Square Foot
Inches of Water (at 4° C)	0.03613	Pounds/Square Inch
International Amperes	0.9998	Amperes (absolute)
International Volts	1.0003	Volts (absolute)
International Volts	1.593×10^{-19}	Joules (absolute)
International Volts	9.654×10^{4}	Joules

J

To Convert:	Multiply by:	To Get:
Joules	9.478×10^{-4}	BTU
Joules	10^{7}	Ergs
Joules	0.7376	Foot-Pounds
Joules	2.389×10^{-4}	Kilogram-Calories
Joules	0.1020	Kilogram-Meters
Joules	2.778×10^{-7}	Kilowatt-Hours
Joules/Centimeter	1.020×10^{4}	Grams
Joules/Centimeter	10^{7}	Dynes
Joules/Centimeter	100.0	Newtons
Joules/Centimeter	723.3	Poundals
Joules/Centimeter	22.48	Pounds

K

To Convert:	Multiply by:	To Get:
Kilderkins	17	Gallons
Kilderkins	77.28	Liters
Kilogram-Calories	3.968	BTU
Kilogram-Calories	3088	Foot-Pounds
Kilogram-Calories	1.560×10^{-3}	Horsepower-Hours
Kilogram-Calories	4186	Joules
Kilogram-Calories	4.186	Kilojoules
Kilogram-Calories	426.9	Kilogram-Meters
Kilogram-Calories	1.163×10^{-3}	Kilowatt-Hours
Kilogram-Meters	9.294×10^{-3}	BTU
Kilogram-Meters	9.804×10^{7}	Ergs
Kilogram-Meters	7.233	Foot-Pounds
Kilogram-Meters	9.804	Joules
Kilogram-Meters	2.342×10^{-3}	Kilogram-Calories
Kilogram-Meters	2.723×10^{-6}	Kilowatt-Hours
Kilograms	980665	Dynes
Kilograms	0.09807	Joules/Centimeter
Kilograms	9.807	Newtons
Kilograms	70.93	Poundals
Kilograms	2.2046226	Pounds
Kilograms	0.0685	Slugs
Kilograms	9.842×10^{-4}	Tons (long)
Kilograms	1.102×10^{-3}	Tons (short)
Kilograms/Cubic Meter	0.06243	Pounds/Cubic Feet
Kilograms/Cubic Meter	3.613×10^{-5}	Pounds/Cubic Inch
Kilograms/Cubic Meter	3.405×10^{-10}	Pounds/Mil Foot
Kilograms/Meter	0.6720	Pounds/Feet
Kilograms/Square Centimeter	980665	Dynes
Kilograms/Square Centimeter	0.9678	Atmospheres
Kilograms/Square Centimeter	32.81	Feet of Water
Kilograms/Square Centimeter	28.96	Inches of Mercury
Kilograms/Square Centimeter	2048	Pounds/Square Foot
Kilograms/Square Centimeter	14.22	Pounds/Square Inch
Kilograms/Square Meter	9.678×10^{-5}	Atmospheres
Kilograms/Square Meter	98.07×10^{-6}	Bars
Kilograms/Square Meter	3.281×10^{-3}	Feet of Water
Kilograms/Square Meter	2.896×10^{-3}	Inches of Mercury
Kilograms/Square Meter	9.806650	Pascals
Kilograms/Square Meter	0.2048	Pounds/Square Foot
Kilograms/Square Meter	1.422×10^{-3}	Pounds/Square Inch
Kilograms/Square Millimeter	10^{6}	Kilograms/Square Meter
Kilolines	1000.0	Maxwells
Kilometers	3281	Feet
Kilometers	3.937×10^{4}	Inches
Kilometers	0.621371	Miles
Kilometers	1094	Yards
Kilometers/Hour	27.78	Centimeters/Second
Kilometers/Hour	54.68	Feet/Minute
Kilometers/Hour	0.9113	Feet/Second
Kilometers/Hour	0.5396	Knots
Kilometers/Hour	16.67	Meters/Minute
Kilometers/Liter	2.3521458	Miles/Gallon (US.)
Kilometers/Liter	2.8248094	Miles/Gallon (Imp.)
Kilowatts	56.92	BTU/Minute
Kilowatts	44253.7	Foot-Pounds/Minute
Kilowatts	736.7	Foot-Pounds/Second
Kilowatts	1.341003	Horsepower
Kilowatts	14.34	Kilogram-Calories/Minute
Kilowatt-Hours	3413.10	BTU
Kilowatt-Hours	3.60×10^{13}	Ergs
Kilowatt-Hours	2.656×10^{6}	Foot-Pounds
Kilowatt-Hours	859850	Gram-Calories
Kilowatt-Hours	1.341	Horsepower-Hours
Kilowatt-Hours	3.6×10^{6}	Joules
Kilowatt-Hours	3.671×10^{5}	Kilogram-Meters
Kilowatt-Hours	3.53	Lbs. of Water evap'd at 212F
Kilowatt-Hours	22.75	"" raised from 62 to 212F
Knots	6080	Feet/Hour
Knots	1.689	Feet/Second
Knots	1.8532	Kilometers/Hour
Knots	1.151	Statute Miles/Hour
Knots	2027	Yards/Hour

L

To Convert:	Multiply by:	To Get:
Leagues (International nautical)	5.556	Kilometers
Leagues (UK nautical)	5.559552	Kilometers
Leagues (US. nautical)	4.828032	Kilometers
Leagues	15,840	Feet
Leagues	3	Miles (approx.)
Leagues	5280	Yards
Legal Subdivisions (Cdn.)	40	Acres
Legal Subdivisions (Cdn.)	0.1618742	Square Kilometers
Light Years	9.46091×10^{12}	Kilometers
Light Years	5.9×10^{12}	Miles
Lines/Square Centimeter	1.0	Gausses
Lines/Square Inch	0.1550	Gausses
Lines/Square Inch	1.550×10^{-9}	Webers/Square Centimeter
Lines/Square Inch	10^{-8}	Webers/Square Inch
Lines/Square Inch	1.550×10^{-5}	Webers/Square Meter
Links (Engineers's)	0.010	Chains
Links (Engineers's)	20.1168	Centimeters

To Convert:	Multiply by:	To Get:
Links (Engineers's)	12.0	Inches
Links (Surveyors's)	7.92	Inches
Liters	0.02838	Bushels (US. dry)
Liters	1000	Cubic Centimeters (cc.)
Liters	0.03531	Cubic Feet
Liters	61.025	Cubic Inches
Liters	1.308×10^{-3}	Cubic Yards
Liters	0.2642	Gallons (US. liquid)
Liters	0.21999	Gallons (Imp. liquid)
Liters	2.1133	Pints (US. liquid)
Liters	1.75969	Pints (Imp. liquid)
Liters	1.0567	Quarts (US. liquid)
Liters	0.87988	Quarts (Imp. liquid)
Liters	0.908	Quarts (dry)
Liters/Minute	5.885×10^{-4}	Cubic Feet/Second
Liters/Minute	4.4033×10^{-3}	Gallons (US.)/Second
Liters/Minute	3.6665×10^{-3}	Gallons (Imp.)/Second
Lumens	0.07958	Spherical Candle Power
Lumens	0.001496	Watts
Lumens/Square Foot	1.0	Foot Candles
Lumens/Square Foot	10.76	Lumens/Square Meter
Lux	0.0929	Foot Candles
	M	
Maxwells	0.001	Kilolines
Maxwells	10^{-8}0.001	Webers
Megalines	10^{6}	Maxwells
Megohms	10^{12}	Microhms
Meters	3.2808399	Feet
Meters	39.37	Inches
Meters	5.396×10^{-4}	Nautical Miles
Meters	6.214×10^{-4}	Statute Miles
Meters	1.0936133	Yards
Meters	1.179	Varas
Meters/Minute	0.05468	Feet/Second
Meters/Minute	0.06	Kilometers/Hour
Meters/Minute	0.03238	Knots
Meters/Minute	0.03728	Miles/Hour
Meters/Second	196.8	Feet/Minute
Meters/Second	3.6	Kilometers/Hour
Meters/Second	2.2369363	Miles/Hour
Meters/Second	0.03728	Miles/Minute
Meter-Kilograms	9.807×10^{7}	Centimeter-Dynes
Meter-Kilograms	10^{5}	Centimeter-Grams
Meter-Kilograms	7.233	Pound-Feet
Microns	10^{-6}	Meters
Miles (UK. Nautical)	1.853184	Kilometers
Miles (US. Nautical)	1.1507794	Miles (Statute)
Miles (US. Nautical)	6,076.11549	Feet
Miles (Statute)	0.8689762	Miles (US. Nautical)
Miles (Statute)	5280	Feet
Miles (Statute)	8	Furlongs
Miles (Statute)	6.336×10^{4}	Inches
Miles (Statute)	1.609344	Kilometers
Miles	1760	Yards
Miles/Hour	44.70	Centimeters/Second
Miles/Hour	88	Feet/Minute
Miles/Hour	1.467	Feet/Second
Miles/Hour	0.8684	Knots
Miles/Hour	26.82	Meters/Minute
Miles/Hour	0.4470	Meters/Second
Miles/Minute	2682	Centimeters/Second
Miles/Minute	88	Feet/Second
Miles/Minute	60	Miles/Hour
Mil-Feet	9.425×10^{-6}	Cubic Inches
Milliers	1000.0	Kilograms
Milligram (mg.)	0.01543236	Grains
Milligrams/Liter	1.0	Parts/Million
Milliliters (ml.)	1.0	Cubic Centimeters
Milliliters	0.271	Drams (liquid)
Milliliters	16.231	Minims
Milliliters	0.061	Cubic Inches
Millimeters	0.0394	Inches
Million Gallons (US.)/Day	1.54723	Cubic Feet/Second
Million Gallons (Imp.)/Day	1.85815	Cubic Feet/Second
Mils	2.540×10^{-3}	Centimeters
Mils	8.333×10^{-5}	Feet
Mils	0.001	Inches
Mils	2.778×10^{-5}	Yards
Miner's Inches	1.5	Cubic Feet/Minute
Minims (British)	0.059192	Cubic Centimeter
Minims (US. liquid)	1.0408	Minims (British)
Minims (US. liquid)	0.061612	Cubic Centimeter
Minutes (angle)	0.01667	Degrees
Minutes (angle)	1.852×10^{-4}	Quadrants
Minutes (angle)	2.909×10^{-4}	Radians
Minutes (angle)	60.0	Seconds
Myriagrams	10.0	Kilograms
Myriameters	10.0	Kilometers
Myriawatts	10.0	Kilowatts
	N	
Nepers	8.686	Decibels
Newtons	0.2248	Pounds
Newtons	10^{5}	Dynes
Newtons/Square Meter	1.0	Pascals

To Convert:	Multiply by:	To Get:
Noggins	1.0	Gills
Noggins	142.1	Milliliters
	O	
Ounces (oz.) (avoirdupois)	16	Drams
Ounces (oz.) (apothecaries')	8	Drams
Ounces (avoirdupois)	437.5	Grains
Ounces (oz. t.) (troy or apothecaries')	480	Grains
Ounces (avoirdupois)	28.350	Grams
Ounces (troy or apothecaries')	31.103	Grams
Ounces (troy or apothecaries')	20.0	Pennyweights
Ounces (avoirdupois)	0.0625	Pounds
Ounces (avoirdupois)	0.9115	Ounces (troy)
Ounces (troy)	1.09714	Ounces (troy)
Ounces (avoirdupois)	2.8349×10^{-5}	Metric Tons
Ounces US. (liquid)	1.041	Ounces British (liquid)
Ounces British (liquid)	0.961	Ounces US. (liquid)
Ounces (fl. oz.) (US.) (liquid)	1.8047	Cubic Inches
Ounces (US.) (liquid)	29.573	Milliliters
Ounces (liquid)	0.125	Cups
Ounces (liquid)	0.0296	Liters
Ounces (British) (liquid)	1.734	Cubic Inches
Ounces (British) (liquid)	28.412	Milliliters
Ounces/Square Inch	4309	Dynes/Square Centimeter
	P	
Pascals	1.0	Newtons/Square Meter
Pascals	0.10197	Kilograms/Square Meter
Pascals	0.020886	Pounds/Square Foot
Pascals	145.03774	Pounds/Square Inch (psi)
Parsecs	19×10^{12}	Miles
Parsecs	3.084×10^{13}	Kilometers
Parts/Million	0.0584	Grains/Gallon (US.)
Parts/Million	0.07016	Grains/Gallon (Imp.)
Parts/Million	8.345	Pounds/Million Gallons (US.)
Pascals (Newtons/Square Meter)	1.45136×10^{-4}	Pounds/Square Inch
Pecks (pk.) (British)	554.6	Cubic Inches
Pecks (British)	9.091901	Liters
Pecks (US.)	0.25	Bushels
Pecks (US.)	537.605	Cubic Inches
Pecks (US.)	8.809582	Liters
Pecks	16	Pints
Pecks	8	Quarts
Pennyweights (dwt.) (troy)	24.0	Grains
Pennyweights (troy)	1.55517	Grams
Pennyweights (troy)	0.05	Ounces (troy)
Pennyweights (troy)	4.1667×10^{-3}	Pounds (troy)
Perch (French area measure)	34.18894	Square Meters
Petrograds (sawn timber)	165.0	Cubic Feet
Petrograds (sawn timber)	4.67228	Cubic Meters
Picas (typography)	0.16667 (1/6)	Inches
Picas	0.4233	Centimeters
Pints (liquid)	473.2	Cubic Centimeters
Pints (liquid)	28.875	Cubic Inches
Pints (liquid)	2	Cups
Pints (liquid)	128	Fluid Drams
Pints (liquid)	16	Fluid Ounces
Pints (liquid)	4	Gills
Pints (liquid)	0.4732	Liters
Pints (dry)	33.600	Cubic Inches
Pints (dry)	0.5510	Liters
Planck's Quantum	6.624×10^{-27}	Erg-Seconds
Points (typography)	0.08333 (1/12)	Picas
Poise	1.00	Grams/Centimeter-Second
Poundals	13826	Dynes
Poundals	14.10	Grams
Poundals	0.1383	Newtons (Joules/Meter)
Poundals	0.01410	Kilograms
Poundals	0.03108	Pounds
Pound-Feet	1.356×10^{7}	Centimeter-Dynes
Pound-Feet	13825	Centimeter-Grams
Pound-Feet	0.13825	Meter-Kilograms
Pounds (lb.) (avoirdupois)	16	Ounces (oz.) (avoirdupois)
Pounds (avoirdupois)	14.5833	Ounces (troy)
Pounds (avoirdupois)	1.21528	Pounds (troy)
Pounds (lb. t.) (troy)	12	Ounces (oz. t.) (troy)
Pounds (troy)	13.1657	Ounces (avoirdupois)
Pounds (troy)	0.82286	Pounds (avoirdupois)
Pounds (avoirdupois)	256	Drams
Pounds (avoirdupois)	7000	Grains
Pounds (avoirdupois)	453.592370	Grams
Pounds (avoirdupois)	4.448	Newtons (Joules/Meter)
Pounds (avoirdupois)	32.17	Poundals
Pounds (avoirdupois)	0.0005	Short Tons
Pounds (troy)	5760	Grains
Pounds (troy)	373.24177	Grams
Pounds (troy)	240.0	Pennyweights (troy)
Pounds (troy)	3.6735×10^{-4}	Tons (long)
Pounds (troy)	3.7324×10^{-4}	Tons (metric)
Pounds (troy)	4.1143×10^{-4}	Tons (short)
Pounds/Cubic Feet	0.01602	Grams/Cubic Centimeter
Pounds/Cubic Feet	5.787×10^{-4}	Pounds/Cubic Inch
Pounds/Cubic Feet	5.456×10^{-9}	Pounds/Mil-Foot
Pounds/Cubic Inch	1728	Pound/Cubic Foot
Pounds/Foot	1.488	Kilograms/Meter
Pounds/Inch	178.6	Grams/Centimeter

To Convert:	Multiply by:	To Get:
Pounds/Mil-Foot	2.306×10^{6}	Grams/Cubic Centimeter
Pounds/Square Foot	4.725×10^{-4}	Atmospheres
Pounds/Square Foot	0.01602	Feet of Water
Pounds/Square Foot	0.01414	Inches of Mercury
Pounds/Square Foot	4.882	Kilograms/Square Meter
Pounds/Square Foot	47.88026	Pascals
Pounds/Square Foot	6.944×10^{-3}	Pounds/Square Inch
Pounds/Square Inch	0.06804	Atmospheres
Pounds/Square Inch	2.307	Feet of Water
Pounds/Square Inch	2.036	Inches of Mercury
Pounds/Square Inch	703.1	Kilograms/Square Meter
Pounds/Square Inch	6894.757	Pascals
Pounds/Square Inch	144.0	Pounds/Square Foot
Pounds of Water	0.0160179	Cubic Feet
Pounds of Water	27.68	Cubic Inches
Pounds of Water	0.1198	Gallons (US.)
Pounds of Water	0.09975	Gallons (Imp.)
Pounds of Water/Minute	2.670×10^{-4}	Cubic Feet/Second
Q		
Quadrants (angle)	90.0	Degrees
Quadrants (angle)	5400.0	Minutes
Quadrants (angle)	1.571	Radians
Quadrants (angle)	3.24×10^{5}	Seconds
Quarters	12.701	Kilograms
Quarters	2.0	Stones
Quarts (qt.) (liquid)	32	Ounces
Quarts (liquid)	256	Drams
Quarts (liquid)	0.25	Gallons
Quarts US. (dry)	0.969	Quarts British
Quarts British (dry)	1.032	Quarts US.
Quarts US. (liquid)	0.833	Quarts British
Quarts British (liquid)	1.201	Quarts US.
Quarts British	69.354	Cubic Inches
Quarts (US.) (dry)	67.201	Cubic Inches
Quarts (US.) (dry)	1.101	Liters
Quarts (US.) (liquid)	0.03342	Cubic Feet
Quarts (US.) (liquid)	57.75	Cubic Inches
Quarts (US.) (liquid)	946.4	Cubic Centimeters
Quarts (US.) (liquid)	1.238×10^{-3}	Cubic Yards
Quarts (US.) (liquid)	0.9463	Liters
R		
Radians	57.2958 (or $180/\pi$)	Degrees
Radians	3438	Minutes
Radians	0.6366	Quadrants
Radians	2.063×10^{5}	Seconds
Radians/Second	9.549	Revolutions/Minute
Radians/Second	0.1592	Revolutions/Second
Revolutions	4	Quadrants
Revolutions	6.283	Radians
Revolutions/Minute	6	Degrees/Second
Revolutions/Second	360	Degrees/Second
Revolutions/Second	6.283	Radians/Second
Rods (Pole or Perch)	0.25	Chains (Gunters)
Rods (Pole or Perch)	16.5	Feet
Rods (Pole or Perch)	5.029	Meters
Rods (Pole or Perch)	5.5	Yards
Roods	0.1011714	Hectares
Roods	1210.0	Square Yards
S		
Scruples (s. ap.)	20	Grains
Scruples	1.296	Grams
Seconds (angle)	2.778×10^{-4}	Degrees
Seconds (angle)	0.01667	Minutes
Seconds (angle)	3.087×10^{-6}	Quadrants
Seconds (angle)	4.8481×10^{-6}	Radians
Sections	640	Acres
Sections	1.0	Square Miles
Sections	2.589988	Square Kilometers
Slugs	14.59	Kilograms
Slugs	32.17	Pounds
Slugs	12.57	Steradians
Square Centimeters	1.973×10^{5}	Circular Mils
Square Centimeters	0.001076	Square Feet
Square Centimeters	3.861×10^{-11}	Square Miles
Square Centimeters	0.1550	Square Inches
Square Centimeters	1.196×10^{-4}	Square Yards
Square Feet	2.2957×10^{-5}	Acres
Square Feet	1.833×10^{8}	Circular Mils
Square Feet	929.0304	Square Centimeters
Square Feet	144	Square Inches
Square Feet	3.5870×10^{-8}	Square Miles
Square Feet	9.290×10^{4}	Square Millimeters
Square Feet	0.1111	Square Yards
Square Feet (French measure)	105.521	Square Centimeters
Square Inches	1.273×10^{6}	Circular Mils
Square Inches	6.4516	Square Centimeters
Square Inches	0.0069	Square Feet
Square Inches	10^{6}	Square Mils
Square Inches	7.716×10^{-4}	Square Yards
Square Kilometers	247.1	Acres
Square Kilometers	10^{10}	Square Centimeters
Square Kilometers	1.0764×10^{7}	Square Feet
Square Kilometers	1.550×10^{9}	Square Inches
Square Kilometers	0.3861	Square Miles
Square Kilometers	1.1960×10^{6}	Square Yards
Square Meters	2.471×10^{-4}	Acres
Square Meters	10.764	Square Feet
Square Meters	1550.0	Square Inches
Square Meters	3.861×10^{-7}	Square Miles
Square Meters	1.1960	Square Yards
Square Miles	640	Acres
Square Miles	27.88×10^{6}	Square Feet
Square Miles	2.589998	Square Kilometers
Square Miles	3.0976×10^{6}	Square Yards
Square Millimeters	1973.0	Circular Mils
Square Millimeters	0.00153	Square Inches
Square Mils	1.273	Circular Mils
Square Mils	6.452×10^{-6}	Square Centimeters
Square Mils	10^{-6}	Square Inches
Square Yards	2.066×10^{-4}	Acres
Square Yards	8361.0	Square Centimeters
Square Yards	9	Square Feet
Square Yards	1296	Square Inches
Square Yards	0.8361274	Square Meters
Square Yards	3.2283×10^{-7}	Square Miles
Stones	6.3503	Kilograms
Stones	14.0	Pounds
T		
Tablespoons	4	Drams (liquid)
Tablespoons	0.5	Ounces (liquid)
Tablespoons	3	Teaspoons
Tablespoons	14.21	Milliliters
Tablespoons (Cdn. Hospital)	15.0	Milliliters
Tablespoons (UK)	17.8	Milliliters
Tablespoons (US.)	14.8	Milliliters
Teaspoons	4.74	Milliliters
Teaspoons	0.16667	Ounces (liquid avoirdupois)
Teaspoons (Cdn. Hospitals)	5.0	Milliliters
Teaspoons (UK.)	5.92	Milliliters
Teaspoons (US.)	4.93	Milliliters
Tons (gross tn.) (gross or long)	1016.0	Kilograms
Tons (gross or long)	2240	Pounds
Tons (gross or long)	1.120	Tons (net or short)
Tons (gross or long)	1.016	Tons (metric)
Tons (tonne or t.) (metric)	1000	Kilograms
Tons (metric)	0.984	Tons (gross or long)
Tons (metric)	1.1023113	Tons (net or short)
Tons (metric)	2204.623	Pounds
Tons (tn. or net tn.) (short or net)	2000	Pounds
Tons (short or net)	907.1848	Kilograms
Tons (short or net)	32000.0	Ounces (avoirdupois)
Tons (short or net)	29166.66	Ounces (troy)
Tons (short or net)	2430.56	Pounds (troy)
Tons (short or net)	0.89286	Tons (long or gross)
Tons (short or net)	0.90718	Tons (metric)
Tons (short or net)/Square Foot	9765.0	Kilograms/Square Meter
Tons of Water/24 Hours	83.333	Pounds of Water/Hour
Tons of Water/24 Hours	0.16643	Gallons (US.)/Minute
Tons of Water/24 Hours	0.13858	Gallons (Imp.)/Minute
Tons of Water/24 Hours	1.3349	Cubic Feet/Hour
Townships	36.0	Sections
Townships	93.23957	Square Kilometers
V		
Volts (absolute)	0.003336	Statvolts
Volts (absolute)	1.602×10^{-19}	Joules
Volts/Inch	0.39370	Volts/Centimeter
W		
Watts	3.4129	BTU (mean)/Hour
Watts	0.056884	BTU (mean)/Minute
Watts	107.0	Ergs/Second
Watts	44.27	Foot-Pounds/Minute
Watts	0.7378	Foot-Pounds/Second
Watts	0.001341	Horsepower
Watts	0.001360	Horsepower (metric)
Watts	1.0	Joules/Second
Watts	0.01433	Kilogram Calories/Minute
Watts (International)	1.0002	Watts (absolute)
Watt-Hours	3.6×10^{10}	Ergs
Watt-Hours	2656	Foot-pounds
Watt-Hours	859.85	Gram-Calories
Watt-Hours	0.001341	Horsepower-Hours
Watt-Hours	367.2	Kilogram-Meters
Webers	10^{8}	Maxwells
Webers	10^{5}	Kilolines
Webers/Square Inch	1.550×10^{7}	Gausses
Webers/Square Inch	10^{8}	Lines/Square Inch
Webers/Square Inch	0.1550	Webers/Square Centimeter
Weber/Square Meter	10^{4}	Gausses
Weber/Square Meter	6.452×10^{4}	Gausses
Webers/Square Meter	10^{-4}	Webers/Square Centimeter
Webers/Square Meter	6.452×10^{-4}	Webers/Square Inch
Y		
Yards	91.44	Centimeters
Yards	4.934×10^{-4}	Miles (nautical)
Yards	5.682×10^{-4}	Miles (statute)

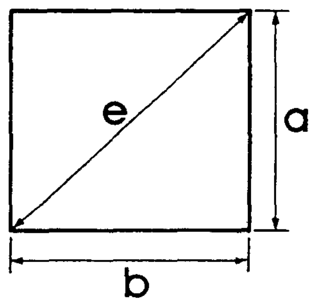
Geometric Areas and Volumes

Nomenclature:

A – Total Area
A_B – Area of Base
A_L – Area of Lateral Surfaces
A_T – Area of Top Section
a,b,c,d – Length of Sides
e,f – Angular Lengths
h,H – Vertical Height
l,L – Arc Length
p – Perimeter
p_B – Perimeter of Base
r, r_1, r_2 – Radii
V – Volume

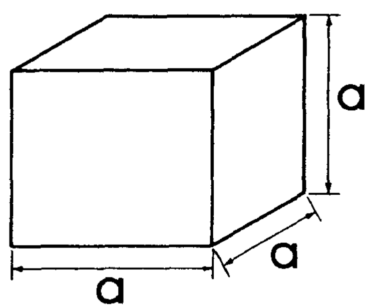
Square

a=b
p=4*a
A=a*a
=.5*e*e
e=a*sqr(2)
=a*1.414



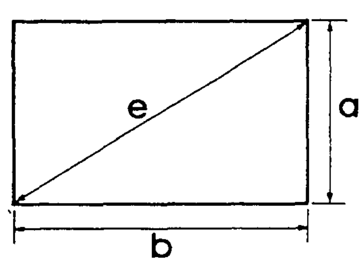
Cube

A=6*a*a
V=a↑3

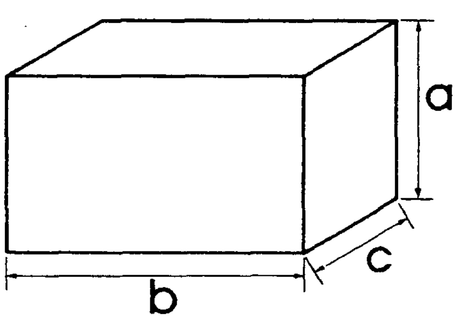


Rectangle

p=2*(a+b)
e=sqr(a*a+b*b)
a=sqr(e*e–b*b)
A=a*b



Parallelopiped

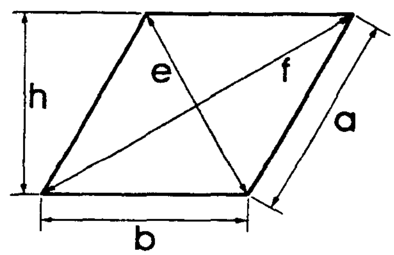


A=2*(a*b+a*c+b*c)
V=a*b*c

Rhombus

(Sides Equal and Parallel)

a=b
p=4*a=4*b
e*e+f*f=4*a*a
A=ah
=e*f/2



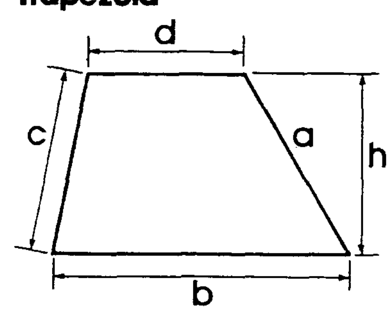
Parallelogram or Rhomboid

(Sides Parallel but Not Equal)

p=2*(a+b)
e*e+f*f=2*(a*a+b*b)
A=ah

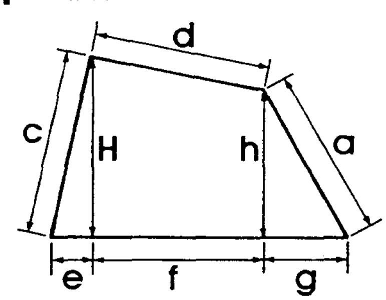
Trapezoid

p=a+b+c+d
A=h*(d+b)/2



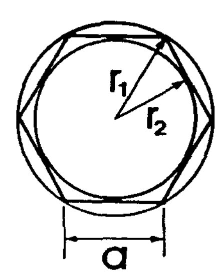
Trapezium

p=a+d+c+e+f+g
A=((H+h)*f+e*H+g*h)/2



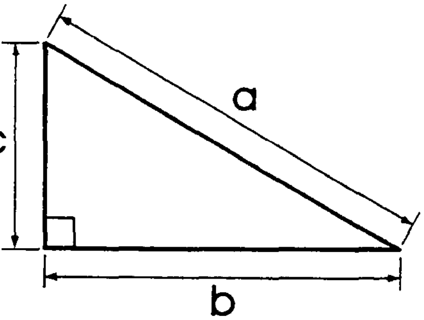
n-Sided Regular Polygon

p=n*a
a=2*sqr(r_1*r_1–r_2*r_2)
A=n*a*r_2/2
=n*a/2*sqr(r_1*r_1–a*a/4)
=n* area of each triangle



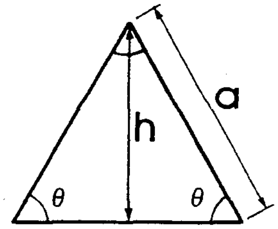
Right Angled Triangle

$p=a+b+c$
$a=sqr(b*b+c*c)$
$b=sqr(a*a-c*c)$
$c=sqr(a*a-b*b)$
$A=b*c/2$



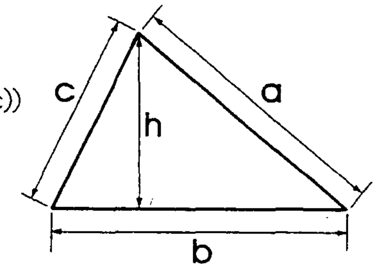
Equilateral Triangle

$p=3*a$
$h=a/2*sqr(3)$
$=a*.8666$
$A=a*a*sqr(3)/4$
$=a*a*.4333$

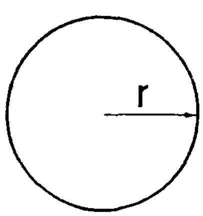


General or Oblique Angled Triangle

$p=a+b+c$
$h=2/b*sqr((s*(s-a)*(s-b)*(s-c))$
where $s=(a+b+c)/2$
$A=b*h/2$
or $A=sqr((s*(s-a)*(s-b)*(s-c))$

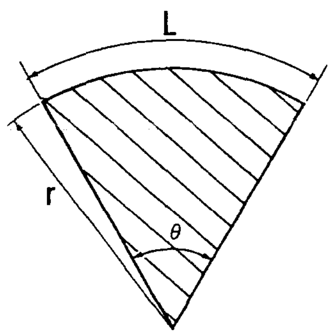


Circle



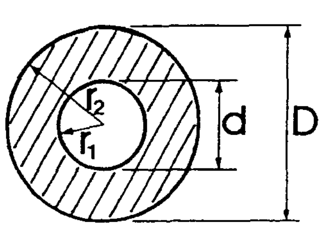
$A=\pi*r*r$
$p=2*\pi*r$

Sector of a Circle



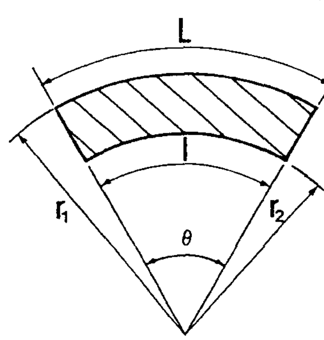
$L=\pi*r*\theta/180$
$=2*A/r$
$A=\pi*\theta*r*r/360$
$=L*r/2$

Hollow Circle or Annulus



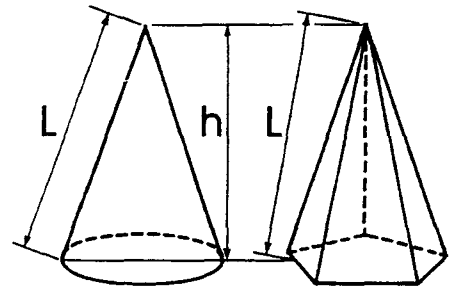
$A=\pi/4*(D*D-d*d)$
$=\pi*(r_2*r_2-r_1*r_1)$
$=\pi/2*(d+D)*(r_2-r_1)$
$=\pi*(r_1+r_2)*(r_2-r_1)$

Sector of a Hollow Circle



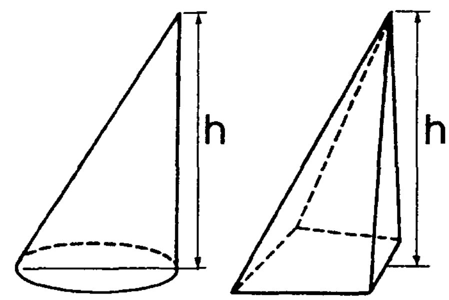
$A=\pi*\theta*(r_2*r_2-r_1*r_1)/360$
$A=(r_1-r_2)*(l+L)/2$

Cone or Pyramid (Right Regular)



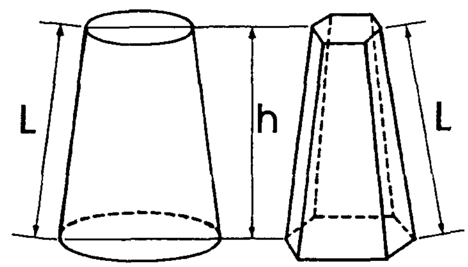
$V=A_B*h/3$
where A_B = area of base
Lateral surface $=p_B*L/2$
where p_B = perimeter of base
$A=\pi*r*sqr(r*r+h*h)+\pi*r*r$

Cone or Pyramid (General)



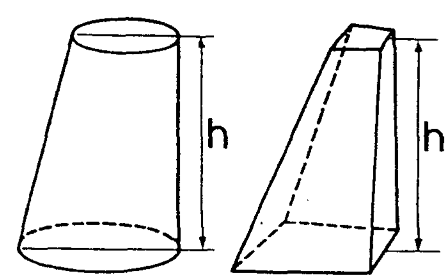
$V=A_B*h/3$
where A_B = area of base

Frustum of a Cone (Right Regular)



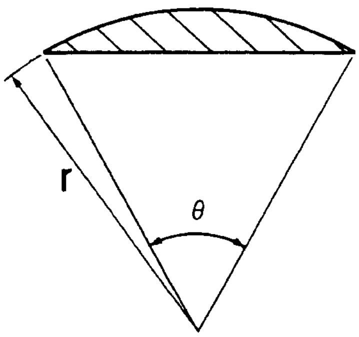
$V=h*(A_B+A_T+sqr(A_B*A_T))/3$
$A_L=L*(p_B+p_T)/2$
$A=A_L+A_B+A_T$
A_B = area of base
A_T = area of top
p_B = perimeter of base
p_T = perimeter of top
A_L = Lateral surface area

Frustum of a Cone (General)



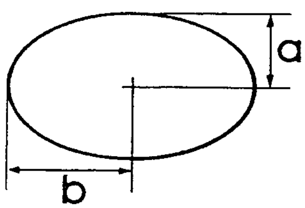
$V=(A_B+A_T+sqr(A_B*A_T))$
where A_B = area of base
and A_T = area of top

Segment of a Circle



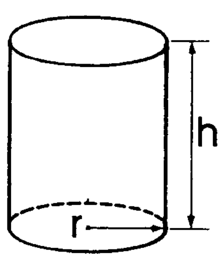
for $\theta<90°$:
$A=r*r*(\pi*\theta/180-\sin(\theta))/2$
for $\theta>90°$:
$A=r*r*(\pi*\theta/180-\sin(180-\theta))/2$

Ellipse



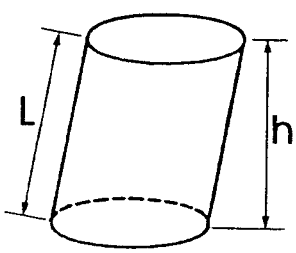
$p\cong\pi*(a+b)$
$p=\pi*(1.5*(a+b)-sqr(a*b))$
(more accurately)
$A=\pi*a*b$

Cylinder (Right Circular)



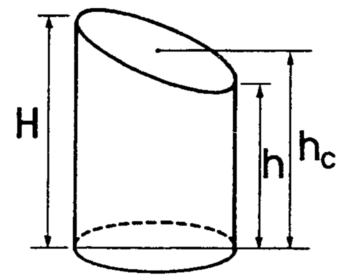
$A_L=2*\pi*r*h$
$A=2*\pi*r*(r+h)$
$V=\pi*r*r*h$

Cylinder (General)



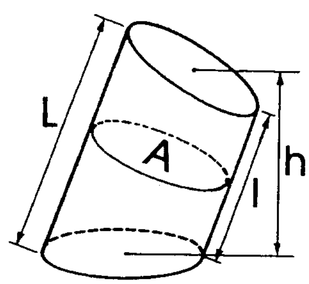
$A_L=p_B*h$
$A=A_L+2*A_B$
$V=A_B*h$
where A_B = area of base $(\pi*r*r)$

Frustum of a Cylinder (Right Circular)



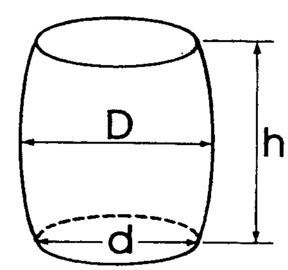
$A_L=\pi*r*(h+H)$
$A_T=\pi*r*sqr(r*r+((h-H)/2)\uparrow 2)$
$A_B=\pi*r*r$
$A=A_L+A_T+A_B$
$V=\pi*r*r*(h+H)/2$

Frustum of a Cylinder (General)



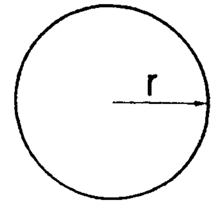
$V=A*(L+l)/2$
$V=A_B*h$

Barrel



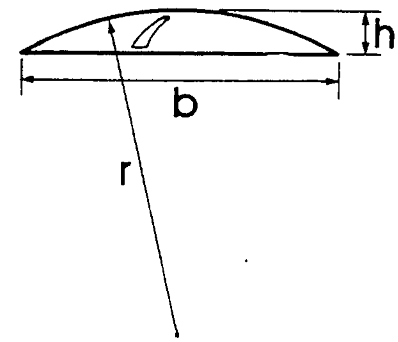
with sides bent to arc of a circle.
$V=\pi*h*(2*D*D+d*d)/12$
with sides bent to arc of a parabola
$V=.209*h*(2*D*D+D*d+.75*d*d)$

Sphere



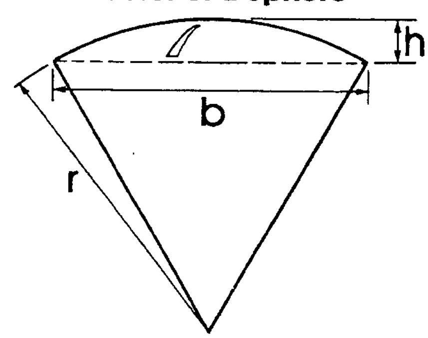
$A=4*\pi*r*r$
$V=4/3*\pi*r\uparrow 3$

Segment of a Sphere



$A=2*\pi*r*h$
or $A=\pi/4*(4*h*h+b*b)$
$V=\pi*h*h*(r-h/3)$
or $V=\pi*h*(b*b/8+h*h/6)$

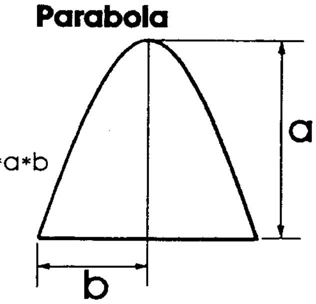
Sector of a Sphere



$A=\pi*r*(2*h+b/2)$
$b=2*sqr(h*(2*r-h))$
$V=2/3*\pi*r*r*h$

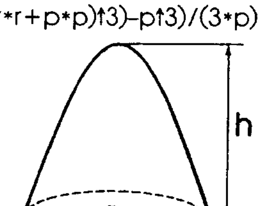
Parabola

A=2/3*a*b

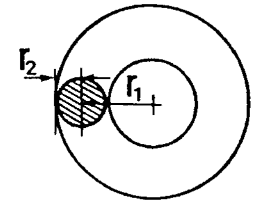


Paraboloid

$A=2*\pi*(sqr((r*r+p*p)\uparrow 3)-p\uparrow 3)/(3*p)$
where:
$p=r*r/(2*h)$
$V=\pi*r*r*h/2$

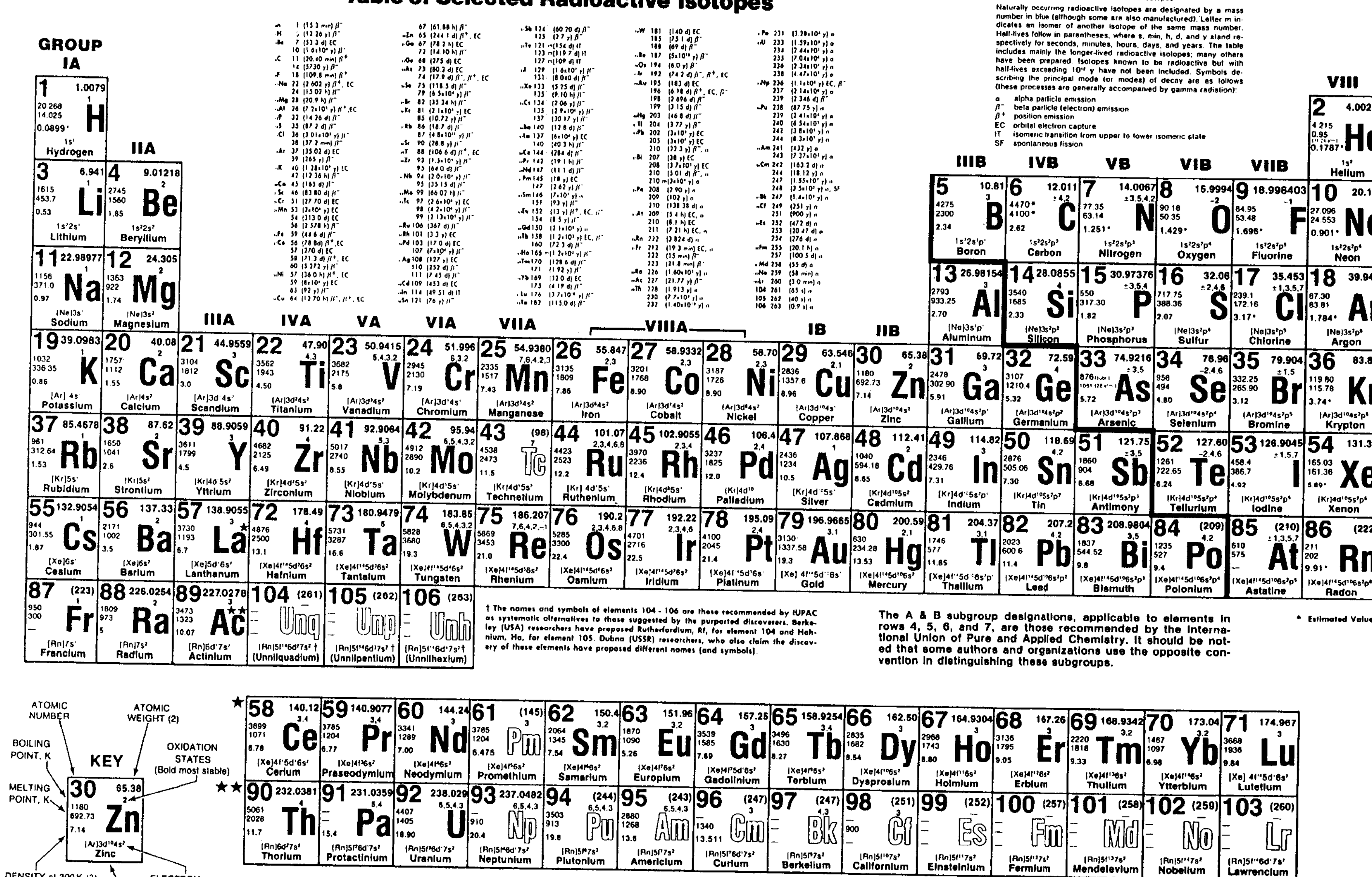


Torus (doughnut)



$A=4*\pi*\pi*r_1*r_2$
$V=2*\pi*\pi*r_2*r_2*r_1$

PERIODIC TABLE OF THE ELEMENTS



NOTES:

(1) Black — solid.
Red — gas.
Blue — liquid.
Outline — synthetically prepared.

(2) Based upon carbon-12. ( ) indicates most stable or best known isotope.

(3) Entries marked with asterisks refer to the gaseous state at 273 K and 1 atm and are given in units of g/l.

Side 1

SARGENT-WELCH SCIENTIFIC COMPANY
7300 NORTH LINDER AVENUE, SKOKIE, ILLINOIS 60077

TABLE OF PERIODIC PROPERTIES OF THE ELEMENTS

Percent Ionic Character of a Single Chemical Bond

Difference in electronegativity	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5	2.6	2.7	2.8	2.9	3.0	3.1	3.2
Percent ionic character %	0.5	1	2	4	6	9	12	15	19	22	26	30	34	39	43	47	51	55	59	63	67	70	74	76	79	82	84	86	88	89	91	92

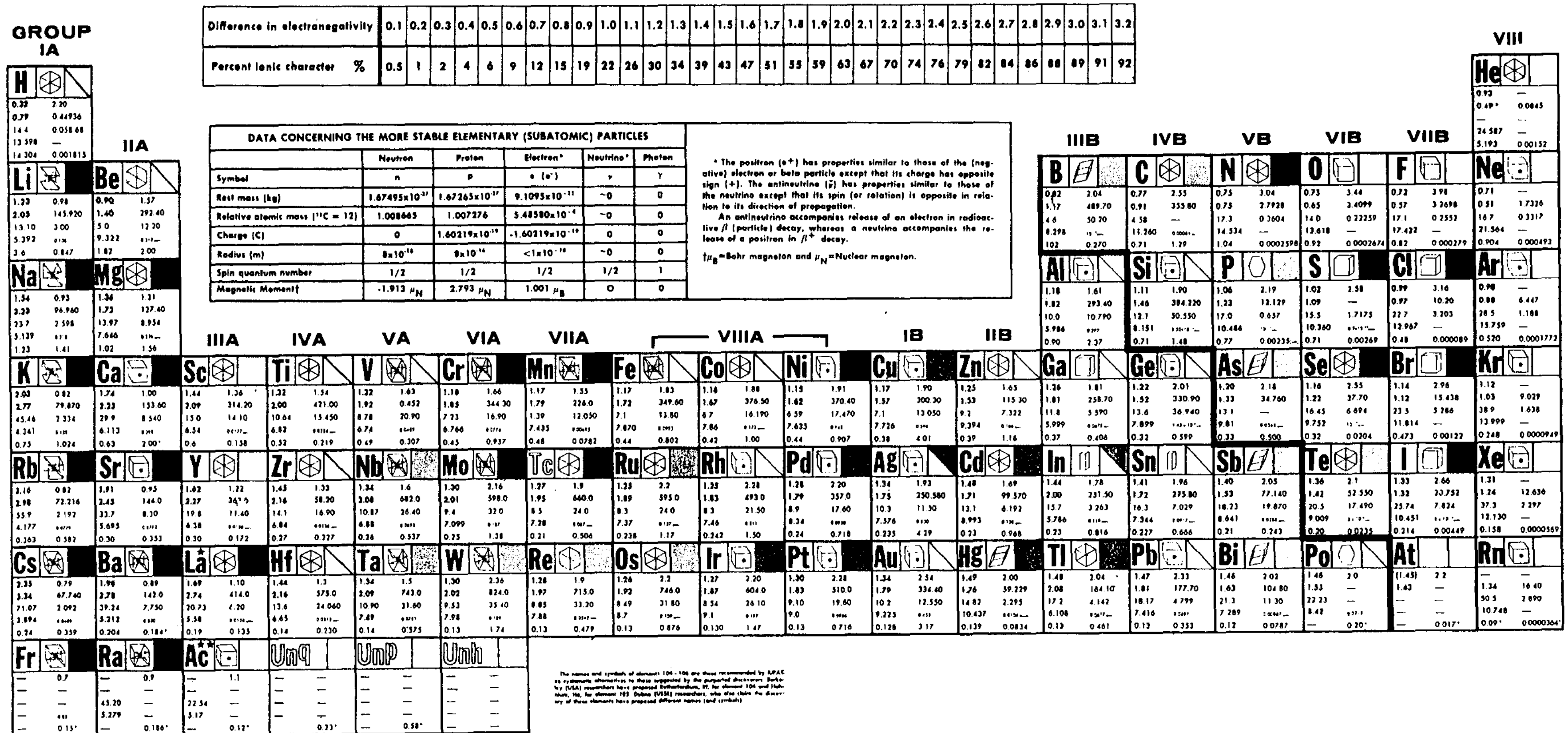
DATA CONCERNING THE MORE STABLE ELEMENTARY (SUBATOMIC) PARTICLES

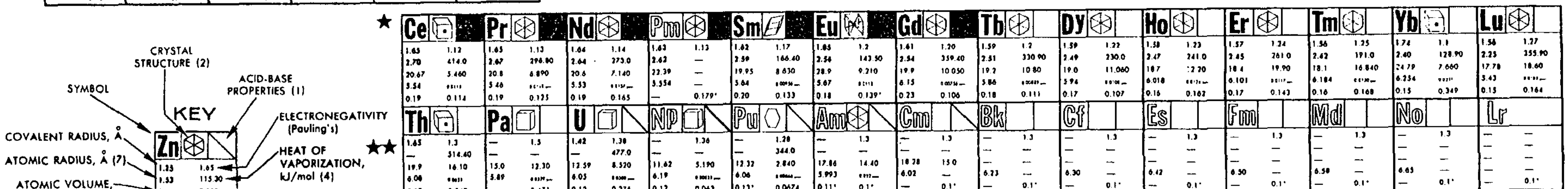
	Neutron	Proton	Electron*	Neutrino*	Photon
Symbol	n	p	e (e^-)	ν	γ
Rest mass (kg)	1.67495×10^{-27}	1.67265×10^{-27}	9.1095×10^{-31}	~0	0
Relative atomic mass (^{12}C = 12)	1.008665	1.007276	5.48580×10^{-4}	~0	0
Charge (C)	0	1.60219×10^{-19}	-1.60219×10^{-19}	0	0
Radius (m)	8×10^{-16}	8×10^{-16}	$< 1 \times 10^{-18}$	~0	0
Spin quantum number	1/2	1/2	1/2	1/2	1
Magnetic Moment†	$-1.913\ \mu_N$	$2.793\ \mu_N$	$1.001\ \mu_B$	0	0

* The positron (e^+) has properties similar to those of the (negative) electron or beta particle except that its charge has opposite sign (+). The antineutrino ($\bar{\nu}$) has properties similar to those of the neutrino except that its spin (or rotation) is opposite in relation to its direction of propagation.

An antineutrino accompanies release of an electron in radioactive β (particle) decay, whereas a neutrino accompanies the release of a positron in β^+ decay.

† μ_B = Bohr magneton and μ_N = Nuclear magneton.





NOTES: (1) For representative oxides (higher valence) of group. Oxide is acidic if color is red, basic if color is blue and amphoteric if both colors are shown. Intensity of color indicates relative strength.

(2) Cubic, face centered; cubic, body centered; cubic; hexagonal; rhombohedral; tetragonal; orthorhombic; monoclinic.

(3) At 300 K (27°C)
(4) At boiling point
(5) At melting point
(6) Generally at 293 K (20°C)
(7) Quantum mechanical value for free atom
(8) From density at 300 K (27°C) for liquid and solid elements; values for gaseous elements refer to liquid state at boiling point

* Copyright 1962 • Copyright 1964 • Copyright 1965 • Copyright 1966 • Copyright 1968 • Copyright 1979 SARGENT-WELCH SCIENTIFIC COMPANY

NSRDS

SARGENT-WELCH
SARGENT-WELCH SCIENTIFIC COMPANY
7300 LINDER AVENUE, SKOKIE, ILLINOIS 60077

Catalog Number S-18806

SIDE 2

Jim Butterfield's Complete C128 Memory Map

A few issues back we published an abridged C128 RAM/ROM map as prepared by Jim Butterfield. At the time we were quite pleased to have the privilege of publication. Although the maps were not in any way complete, they were good enough to start many hungry programmers on their way with the C128.

After many months of careful and very well calculated pestering on our part, Jim has finally consented to allow us to publish his yet unreleased C128 Map. This opportunity comes as a form of prelude to Jim's yet unreleased new version of, "Machine Language For The Commodore 64 And Other Commodore Computers". Jim has carefully re-written it to include the C128, and as is usual with Jim's books, articles, videos, TV shows, etc., etc., etc., his Machine Language book takes the reader by the hand and gently force feeds knowledge without any painful infliction.

Jim's new book is expected to be released in April of 1986, published by Bradey, a division of Simon and Shuster. As with his last Machine Language book, this version will be available most everywhere through many of the major book stores. If after this incredible bit of JB propaganda you remain unmoved, let me assure you that I am not being paid for this, except for a bottle of Steam beer he bought me in San Francisco (for which I paid him back promptly). If ever you get the chance, have a read. . . you will not be disappointed. - RTE

COMMODORE 128 Memory Maps **Jim Butterfield**

These maps apply to the machine when used in the 128K mode. When used in the 64 mode, the machine's map is identical to that of the Commodore 64.

Architecture: "Bank numbers" as used in Basic BANK and the MLM addressing scheme are misleading; in fact, they are more correctly "configuration numbers". Bank 0 shows RAM level 0, which contains work areas and the user's Basic program. Bank 1 also shows RAM, this time (for addresses above hexadecimal 0400) level 1 which contains variables, arrays, and strings. Other "banks" are really configurations, with various types of ROM or I/O overlaying RAM. Thus, bank 15 (the most popular) is ROM and I/O covering RAM bank 0. Bank 14, however, is ROM and the character generator overlaying RAM bank 0. Architecture is set so that addresses below $0400 reference bank 0 only. Other bank switching (more complex than the simplified 16-bank concept) is accomplished via storing a mask to address $FF00, or calling up pre-stored masks by writing to $FF01-FF04.

The Commodore C128 Memory Map as of February 1986

All Banks:

Hex	Decimal	Description
0000	0	I/O directional register
0001	1	I/O port, similar to C64
0002 -0004	2-4	SYS address, MLM registers (SR, PC)
0005 -0009	5-9	SYS, MLM register save (A, X, Y, SR/SP)
000A	10	Scan-quotes flag
000B	11	TAB column save
000C	12	0=LOAD, 1=VERIFY
000D	13	Input buffer pointer/number of subscripts
000E	14	Default DIM flag
000F	15	Type: FF=string; 00=numeric
0010	16	Type: 80=integer; 00=floating point
0011	17	DATA scan/LIST quote/memory flag
0012	18	Subscript/FNx flag
0013	19	0=INPUT;$40=GET;$98=READ
0014	20	ATN sign/Comparison evaluation flag
0015	21	Current I/O prompt flag
0016 -0017	22-23	Integer value
0018	24	Pointer: temporary string stack
0019 -0023	25-35	Stack for temporary strings
0024 -0027	36-39	Utility pointer area
0028 -002C	40-44	Product area for multiplication
002D -002E	45-46	Pointer: start-of-BASIC (for bank 0)
002F -0030	47-48	Pointer: start-of-variables (bank 1)
0031 -0032	49-50	Pointer: start-of-arrays
0033 -0034	51-52	Pointer: end-of-arrays
0035 -0036	53-54	Pointer: string-storage (moving down)
0037 -0038	55-56	Utility string pointer
0039 -003A	57-58	Pointer: limit-of-memory (bank 1)
003B -003C	59-60	Current BASIC line number
003D -003E	61-62	Textpointer: BASIC work point
003F -0040	63-64	Utility Pointer
0041 -0042	65-66	Current DATA line number
0043 -0044	67-68	Current DATA address
0045 -0046	69-70	Input vector
0047 -0048	71-72	Current variable name
0049 -004A	73-74	Current variable address
004B -004C	75-76	Variable pointer for FOR/NEXT
004D -004E	77-78	Y-save; op-save; BASIC pointer save
004F	79	Comparison symbol accumulator
0050 -0055	80-85	Miscellaneous work area, pointers, and so on
0056 -0058	86-88	Jump vector for functions
0059 -0062	89-98	Miscellaneous numeric work area
0063	99	Accum#1: exponent
0064 -0067	100-103	Accum#1: mantissa
0068	104	Accum#1: sign
0069	105	Series evaluation constant pointer
006A -006F	106-111	Accum#2: exponent, and so on
0070	112	Sign comparison, Acc#1 versus #2
0071	113	Accum#1 lo-order (rounding)
0072 -0073	114-115	Cassette buffer len/Series pointer
0074 -0075	116-117	Auto line number increment
0076	118	Graphics flag
0077	119	Color source number
0078 -0079	120-121	Temporary counters
007A -007C	122-124	DS$ descriptor
007D -007E	125-126	BASIC pseudo-stack pointer
007F	127	Flag: 0=direct mode
0080 -0081	128-129	DOS, USING work flags
0082	130	Stack pointer save for errors
0083	131	Graphic color source
0084	132	Multicolor 1 (1)
0085	133	Multicolor 2 (2)
0086	134	Graphic foreground color (13)
0087 -008A	135-138	Graphic scale factors, X & Y
008B -008F	139-143	Graphic work values
0090	144	Status word ST
0091	145	Keyswitch IA: STOP and RVS flags
0092	146	Timing constant for tape
0093	147	Work value, monitor, LOAD/SAVE
0094	148	Serial output: deferred character flag
0095	149	Serial deferred character
0096	150	Cassette work value
0097	151	Register save
0098	152	How many open files
0099	153	Input device, normally 0
009A	154	Output CMD device, normally 3
009B -009C	155-156	Tape parity, output-received flag
009D	157	I/O messages: 192=all, 64=errors, 0=nil
009E -009F	158-159	Tape error pointers
00A0 -00A2	160-162	Jiffy Clock HML
00A3 -00AB	163-171	I/O work bytes
00AC -00AD	172-173	Pointer: tape buffer, scrolling
00AE -00AF	174-175	Tape end adds/End of program
00B0 -00B1	176-177	Tape timing constants
00B2 -00B3	178-179	Pointer: start of tape buffer
00B4 -00B6	180-182	RS-232, Misc work values
00B7	183	Number of characters in file name
00B8	184	Current logical file
00B9	185	Current secondary address
00BA	186	Current device
00BB -00BC	187-188	Pointer to file name
00BD -00C5	189-197	I/O work pointers
00C6 -00C7	198-199	Banks: I/O data, filename
00C8 -00CB	200-203	RS-232 input/output buffer addresses
00CC -00CD	204-205	Keyboard decode pointer (bank 15)
00CE -00CF	206-207	Print string work pointer
00D0	208	Number of characters in keyboard buffer
00D1	209	Number of programmed chars waiting
00D2	210	Programmed key character index
00D3	211	Key shift flag: 0=no shift
00D4	212	Key code: 88 if no key
00D5	213	Key code: 88 if no key
00D6	214	Input from screen/from keyboard
00D7	215	40/80 columns: 0=40 columns
00D8	216	Graphics mode code
00D9	217	Character base: 0=ROM, 4=RAM
00DA-00DF	218-223	Misc work area
00E0 -00E1	224-225	Pointer to screen line/cursor
00E2 -00E3	226-227	Color line pointer
00E4	228	Current screen bottom margin
00E5	229	Current screen top margin
00E6	230	Current screen left margin
00E7	231	Current screen right margin
00E8 -00E9	232-233	Input cursor log (row, column)
00EA	234	End-of-line for input pointer
00EB	235	Position of cursor on screen line
00EC	236	Row where cursor lives
00ED-00EE	237-238	Maximum screen lines, columns
00EF	239	Current I/O character
00F0	240	Previous character printed
00F1	241	Character color
00F2	242	Temporary color save
00F3	243	Screen reverse flag
00F4	244	0=direct cursor; else programmed
00F5	245	Number of INSERTs outstanding
00F6	246	255=Auto Insert enabled
00F7	247	Text mode lockout
00F8	248	0=Scrolling enablled
00F9	249	Bell disable
00FA -00FF	250-255	Not used
0100 -01FF	256-511	Processor stack area
0100 -013E	256-318	Tape error log
0100 -0124	256-292	DOS work area
0125 -0138	293-312	PRINT/USING work area
0200 -02A0	512-672	BASIC input buffer
02A2 -02AE	674-686	Bank peek subroutine
02AF -02BD	687-701	Bank poke subroutine
02BE -02CC	702-716	Bank compare subroutine
02CD -02E2	717-738	JSR to another bank
02E3 -02FB	739-763	JMP to another bank
02FC -02FD	764-765	Function execute hook [4C78]
0300 -0301	768-769	Error message link
0302 -0303	770-771	BASIC warm start link
0304 -0305	772-773	Crunch BASIC tokens link
0306 -0307	774-775	Print tokens link
0308 -0309	776-777	Start new BASIC code link
030A -030B	778-779	Get arithmetic element link
030C -030D	780-781	Crunch FE hook
030E -030F	782-783	List FE hook
0310 -0311	784-785	Execute FE hook
0312 -0313	786-787	Unused
0314 -0315	788-789	IRQ vector [FA65]
0316 -0317	790-791	Break interrupt vector [B003]
0318 -0319	792-793	NMI interrupt vector [FA40]
031A -031B	794-795	OPEN vector [EFBD]

Note: Address values shown in **bold** are corrections since originally published.

031C -031D	796–797	CLOSE vector [F188]
031E -031F	798–799	Set-input vector [F106]
0320 -0321	800–801	Set-output vector [F14C]
0322 -0323	802–803	Restore I/O vector [F226]
0324 -0325	804–805	Input vector [EF06]
0326 -0327	806–807	Output vector [EF79]
0328 -0329	808–809	Test-STOP vector [F66E]
032A -032B	810–811	GET vector [EEEB]
032C -032D	812–813	Abort I/O vector [F222]
032E -032F	814–815	Machine Lang Monitor link
0330 -0331	816–817	LOAD link
0332 -0333	818–819	SAVE link
0334 -0335	820–821	Control code (low) link
0336 -0337	822–832	High ASCII code link
0338 -0339	824–825	ESC sequence link
034A -0353	842–851	Keyboard buffer
0354 -035D	852–861	Tab stop bits
035E -0361	862–865	Line wrap bits
0362 -036B	866–875	Logical file table
036C -0375	876–885	Device number table
0376 -037F	886–895	Secondary address table
0380 -039E	896–926	CHRGET subroutine
0386	902	CHRGOT entry
039F -03AA	927–938	Fetch from RAM bank 0
03AB -03B6	939–950	Fetch from RAM bank 1
03B7 -03BF	951–959	Fetch from RAM bank 1
03C0 -03C8	960–968	Fetch from RAM bank 0
03C9 -03D1	969–977	Fetch from RAM bank 0
03D2 -03D4	978–980	Unused
03D5	981	Current BANK for SYS, PEEK
03D6 -03D9	982–985	INSTR work values
03DA	986	Bank location for string
03DB -03DD	987–989	Sprite work bytes
03DF	991	Accum#1: Overflow
03E0 -03E1	992–993	Sprite work bytes
03E2	994	Graphic/Text backgrounds
03E3	995	Graphic/Multi color log
03F0 -03F6	1008–1014	DMA link code
FF00	65280	MMU configuration register
FF01		Bank 0
FF02		Bank 1
FF03		Bank 14
FF04		Bank 14 over RAM 1
FF01 -FF04	65281-65284	MMU load config registers
Bank 0:		
0400 -07E7	1024–2023	40-column screen memory
07F8 -07FF	2040–2047	Sprite identity area (text)
0800 -09FF	2048–2560	BASIC pseudo-stack
0A0C	2572	CIA 1 interrupt log
0A0D	2573	CIA 1 timer enabled
0A0F -0A17	2575–2583	RS-232 work values
0A18	2584	RS-232 receive pointer
0A19	2585	RS-232 input pointer
0A1A	2586	RS-232 transmit pointer
0A1B	2587	RS-232 send pointer
0A1D -0A1F	2588–2590	Sleep countdown: FFFF = disable
0A20	2592	Keyboard buffer size
0A21	2593	Screen freeze flag
0A22	2594	Key repeat: 128 = all, 64 = none
0A23	2595	Key repeat timing
0A24	**2596**	Key repeat pause
0A25	**2597**	Graphics/text toggle latch
0A26	**2598**	40-col cursor mode
0A27 -0A2A	**2599-2602**	40-col blink values
0A2B	**2603**	80-col cursor mode
0A2C	**2604**	40-col video $D018 image
0A2E -0A2F	2606–2607	80 col pages – screen, color
0A40 -0A5A	2624–2650	40/80 pointer swap $E0–FA
0A60 -0A6D	2656–2669	40/80 data swap $354–361
0AC0	2752	PAT counter
0AC1 -0AC4	2753–2756	ROM Physical Address Table
0B00 -0BBF	2816–3007	Cassette buffer
0BC0 -0BFF	3008–3071	
0C00 -0DFF	3072–3583	RS-232 input, output buffers
0E00 -0FFF	3584–4095	System sprites (56–63)
1000 -1009	4096–4105	Programmed key lengths
100A -10FF	4106–4351	Programmed key definitions
1100 -1130	4352–4400	DOS Command staging area
1131 -116E	4401–4462	Graphics work area
116F	4463	Trace mode: FF = on
1170 -1173	4464–4467	Renumbering pointers
1174 -1177	4468–4471	Directory work pointers
1178 -**1179**	4472–4473	Graphics index
117A -117B	4474–4475	Float-fixed vector [849F]
117C -117D	4476–4477	Fixed-float vector [793C]
117E -11D5	4478–4565	Sprite motion tables (8 x 11 bytes)
11D6 -11E5	4566–4581	Sprite X/Y positions
11E6	4582	Sprite X-high positions
11E7 -11E8	4583–4584	Sprite bump masks (sprite, backgnd)
11E9 -11EA	4585–4586	Light pen values, X and Y
11EB	4587	CHRGEN ROM page, text [D8]
11EC	4588	CHRGEN ROM page, graphics [D0]
11ED	4589	Secondary address for RECORD
11EE -11FF	4590–4607	Unused
1204 -1207	4612–4615	PU characters (,.$)
120B -120C	4619–4620	TRAP address: FFFF if none
1210 -1211	4624–4625	End of Basic (Bank 0)
1212 -1213	4626–4627	Basic program limit [FF00]
1214 -1217	4628–4631	DO work pointers
1218 -121A	4632–4634	USR program jump [7D28]
121B -121F	4635–4639	RND seed value
1222	4642	Sound tempo
122F	4655	Music sequencer
1234 -1237	4660–4663	Note image
1239 -123E	4665–4670	Current env pattern
123F -1270	4671–4720	Envelope tables ..
123F -1248	4671–4680	AD(SR) pattern
1249 -1252	4681–4690	(AD)SR pattern
1253 -125C	4691–4700	Waveform pattern
125D -1266	4701–4710	Pulse width pattern
1267 -1270	4711–4720	Pulse width hi pattern
1271 -1274	4721-**4724**	Note: xx,xx,volume
1275	4725	Previous volume image
1276 -1278	4726–4728	Collision IRQ task table
1279 -127E	4729–4734	Collision IRQ address tables
127F	4735	Collision mask
1280	4736	Collision work value
12B1	4785	PEN work value
1300 -17FF	4864–6143	Unused
1800 -1BFF	6144–7167	Reserved for key functions
1C00 -FBFF	7168–64511	BASIC RAM memory (text)
1C00 -1FF7	7168–8186	Video (color) matrix (hi-res)
1FF8 -1FFF	8187–8191	Sprite identities (hi-res)
2000 -3FFF	8192–16383	Screen memory (hi-res)
4000 -FBFF	16384–64511	BASIC RAM memory (hi-res)
Bank 1:		
0400 -FBFF	1024–64511	Basic variables, arrays, strings
Bank 14: Same as Bank 15, below, except:		
D000 -DFFF	53248–57343	Character generator ROM
Bank 15:		
4000 -CFFF	16384–53247	ROM: BASIC
D000 -D02E	53248–53294	40-col video chip 8564
D400 -D41C	54272–54300	SID sound chip 6581
		Memory Management Unit 8722
D500	54528	MMU primary config register
D501 -D504	54529–54532	MMU preconfig registers
D505 -D506	54533–54534	MMU mode, ram registers
D507 -D50A	54535–54538	MMU page 0, page 1 regs
D600 -D601	54784–54785	80-column CRT contr 8563
10 -11	16–17	X, Y positions
12 -13	18–19	On-chip RAM address
1A	26	Background color
1F	31	On-chip RAM data
D800 -**DBE7**	55296–56295	Color nybbles
DC00 -DC0F	56320–56336	CIA 1 (IRQ) 6526
DD00 -DD0F	56576–56591	CIA 2 (NMI) 6526
DF00 -DF0A	57088–57098	DMA slot
E000 -FEFF	57344–65279	ROM: Kernal
FF05 -FFFF	65285–65535	ROM: Transfer, Jump Table

ROM Map

4000	Basic Entry Jumps
4009	Basic Restart
4023	Basic Cold Start
4045	Set-Up Basic Constants
4112	Chime
417A	Set Preconfig Registers
4189	Registers For $D501
418D	Init Sprite Movement Tabs
419B	Print Startup Message
4251	Set Basic Links
4267	Basic Links
4279	Chrget For $0380
42CE	Get From ($50) Bank 1
42D3	Get From ($3F) Bank 1
42D8	Get From ($52) Bank 1
42DD	Get From ($5C) Bank 0
42E2	Get From ($5C) Bank 1
42E7	Get From ($66) Bank 1
42EC	Get From ($61) Bank 0
42F1	Get From ($70) Bank 0
42F6	Get From ($70) Bank 1
42FB	Get From ($50) Bank 1
4300	Get From ($61) Bank 1
4305	Get From ($24) Bank 0
430A	Crunch Tokens
43E2	Check Keyword Match
4417	Keywords
46FC	Action Vectors
47D8	Function Vectors
4828	Defunct Vectors
4846	Unimplemented Commands
484B	Messages
4A82	Find Message
4B34	Update Continue Pointer
4B3F	Execute/Trace Statement
4BCB	Perform [stop]
4BCD	Perform [end]
4BF7	Setup FN Reference
4C86	Evaluate <or>
4C89	Evaluate <and>
4CB6	Evaluate <compare>
4D2A	Print 'ready'
4D37	Error or Ready
4D3A	Print 'out of memory'
4D3C	Error
4DAF	Break Entry
4DC3	Ready For Basic
4DE2	Handle New Line
4F4F	Rechain Lines
4F82	Reset End-of-Basic
4F93	Receive Input Line
4FAA	Search B-Stack For Match
4FFE	Move B-Stack Down
5017	Check Memory Space
5047	Copy B-Stack Pointer
5050	Set B-Stack Pointer
5059	Move B-Stack Up
5064	Find Basic Line
50A0	Get Fixed Pt Number
50E2	Perform [list]
5123	List Subroutine
51D6	Perform [new]
51F3	Set Up Run
51F8	Perform [clr]
5238	Clear Stack & Work Area
5250	Pudef Characters
5254	Back Up Text Pointer
5262	Perform [return]
528F	Perform [data/bend]
529D	Perform [rem]
52A2	Scan To Next Stmnt
52A5	Scan To Next Line
52C5	Perform [if]
5320	Search/Skip Begin/Bend
537C	Skip String Constant
5391	Perform [else]
53A3	Perform [on]
53C6	Perform [let]
54F6	Check String Location
553A	Perform [print#]
5540	Perform [cmd]
555A	Perform [print]
5600	Print Format Char
5612	Perform [get]
5635	Getkey
5648	Perform [input#]
5662	Perform [input]
569C	Prompt & Input
56A9	Perform [read]
57F4	Perform [next]
587B	Perform [dim]
5885	Perform [sys]
58B4	Perform [tron]
58B7	Perform [troff]
58BD	Perform [rreg]
5901	Assign <mid$>
5975	Perform [auto]
5986	Perform [help]
59AC	Insert Help Marker
59CF	Perform [gosub]
59DB	Perform [goto]
5A15	Undef'd Statement
5A1D	Put Sub To B-Stack
5A3D	Perform [go]
5A60	Perform [cont]
5A9B	Perform [run]
5ACA	Perform [restore]
5AF0	Keywords To Renumber
5AF8	Perform [renumber]
5BAE	Renumber - Continued
5BFB	Renumber Scan
5D19	Convert Line Number
5D68	Get Renumber Start
5D75	Count Off Lines
5D89	Add Renumber Inc
5D99	Scan Ahead
5DA7	Set Up Block Move
5DC6	Block Move Down
5DDF	Block Move Up
5DEE	Check Block Limit
5DF9	Perform [for]
5E87	Perform [delete]
5EFB	Get Line Number Range
5F34	Perform [pudef]
5F4D	Perform [trap]
5F62	Perform [resume]
5FB7	Reinstate Trap Point
5FD8	Syntax Exit
5FDB	Print 'can't resume'
5FE0	Perform [do]
6039	Perform [exit]
608A	Perform [loop]
60B4	Print 'loop not found'
60B7	Print 'loop without do'
60DB	Eval While/Until Argument
60E1	Define Programmed Key
610A	Perform [key]
61A8	Perform [paint]
627C	Check Painting Split
62B7	Perform [box]
642B	Perform [sshape]
658D	Perform [gshape]
668E	Perform [circle]
6750	Draw Circle
6797	Perform [draw]
67D7	Perform [char]
6955	Perform [locate]
6960	Perform [scale]
69E2	Perform [color]
6A4C	Color Codes
6A5C	Log Current Colors
6A79	Perform [scnclr]
6B06	Fill Memory Page
6B17	Set Screen Color
6B30	Clear Hi-Res Screen
6B5A	Perform [graphic]
6BC9	Perform [bank]
6BD7	Perform [sleep]
6C09	Multiply Sleep Time
6C2D	Perform [wait]
6C4F	Perform [sprite]
6CB3	Bit Masks
6CC6	Perform [movspr]
6DE1	Perform [play]
6E02	Analyze Play Character
6EB2	Set SID Sound
6EFD	Play Error
6F03	Dotted Note
6F07	Note Length Char
6F1E	Note A-G

6F52	.. votxum ..
6F69	Sharp
6F6C	Flat
6F78	Rest
6FD7	Perform [tempo]
6FE4	Voice Times Two
6FE7	Length Characters
6FEC	Command Characters
702F	Chime Seq
7039	SID Voice Steps
7046	Perform [filter]
70C1	Perform [envelope]
7164	Perform [collision]
7190	Perform [sprcolor]
71B6	Perform [width]
71C5	Perform [vol]
71EC	Perform [sound]
72CC	Perform [window]
7335	Perform [boot]
7372	Perform [sprdef]
7691	Sprite Vectors
76EC	Perform [sprsav]
77B3	Perform [fast]
77C4	Perform [slow]
77D7	Type Match Check
77DA	Confirm Numeric
77DD	Confirm String
77E7	Print 'type mismatch'
77EA	Print 'formula too complex'
77EF	Evaluate Expression
78D7	Evaluate Item
793C	Fixed-Float
7950	Eval Within Parens
795C	Check Comma
796C	Syntax Error
7978	Search For Variable
7A85	Unpack RAM1 to FAC#1
7AAF	Locate Variable
7B3C	Check Alphabetic
7B46	Create Variable
7CAB	Set Up Array
7D25	Print 'bad subscript'
7D28	Print 'illegal quantity'
7E3E	Compute Array Size
7E71	Array Pointer Subrtn
8000	Evaluate <fre>
8020	Decrypt Message
804A	Evaluate <val>
8052	String to Float
8076	Evaluate <dec>
80C5	Evaluate <peek>
80E5	Perform [poke]
80F6	Evaluate <err$>
8139	Swap .x With .y
8142	Evaluate <hex$>
816B	Byte to Hex
8182	Evaluate <rgr>
818C	Get Graphics Mode
819B	Evaluate <rclr>
8203	Evaluate <joy>
824D	Evaluate <pot>
82AE	Evaluate <pen>
82FA	Evaluate <pointer>
831E	Evaluate <rsprite>
8361	Evaluate <rspcolor>
837C	Evaluate <bump>
8397	Evaluate <rspos>
83E1	Evaluate <xor>
8407	Evaluate <rwindow>
8434	Evaluate <rnd>
8490	Rnd Multiplier
849A	Value 32768
849F	Float-Fixed Unsigned
84A7	Evaluate Fixed Number
84AD	Float-Fixed Signed
84C9	Float (.y,.a)
84D0	Evaluate <pos>
84D9	Check Direct
84DD	Print 'illegal direct'
84E0	Print 'undef'd function'
84E5	Set Up 16 Bit Fix-Float
84F5	Print 'direct mode only'
84FA	Perform [def]
8528	Check FN Syntax
853B	Perform [fn]
85AE	Evaluate <str$>
85BF	Evaluate <chr$>
85D6	Evaluate <left$>
860A	Evaluate <right$>
861C	Evaluate <mid$>
864D	Pull String Parameters
8668	Evaluate <len>
866E	Exit String Mode
8677	Evaluate <asc>
8688	Calc String Vector
869A	Set Up String
874E	Build String to Memory
877B	Evaluate String
87E0	Clean Descriptor Stack
87F1	Input Byte Parameter
8803	Params For Poke/Wait
8815	Float/Fixed
882E	Subtract From Memory
8831	Evaluate <subtract>
8845	Add Memory
8848	Evaluate <add>
8917	Trim FAC#1 Left
894E	Round Up FAC#1
895D	Print 'overflow'
899C	Log Series
89CA	Evaluate <log>
8A0E	Add 0.5
8A24	Multiply By Memory
8A27	Evaluate <multiply>
8A89	Unpack ROM to FAC#2
8AB4	Unpack RAM1 to FAC#2
8AE3	Adjust FAC#1/#2
8B17	Multiply By 10
8B2E	+ 10
8B33	Print 'division by zero'
8B38	Divide By 10
8B49	Divide Into Memory
8B4C	Evaluate <divide>
8BD4	Unpack ROM to FAC#1
8BF9	Pack FAC#1 to $5E
8BFC	Pack FAC#1 to $59
8C00	Pack FAC#1 to RAM1
8C28	FAC#2 to FAC#1
8C38	FAC#1 to FAC#2
8C47	Round FAC#1
8C57	Get Sign
8C65	Evaluate <sgn>
8C68	Byte Fixed-Float
8C75	Fixed-Float
8C84	Evaluate <abs>
8C87	Compare FAC#1 to Memory
8CC7	Float-Fixed
8CFB	Evaluate <int>
8D22	String to FAC#1
8DB0	Get Ascii Digit
8E17	Conversion Values
8E26	Print 'in'. . .
8E32	Print Integer
8E42	Float to Ascii
8F76	+ 0.5
8F7B	Decimal Constants
8F9F	TI Constants
8FB7	Evaluate <sqr>
8FBE	Raise to Memory Power
8FC1	Evaluate <power>
8FFA	Evaluate <negate>
9005	Exp Series
9033	Evaluate <exp>
90D0	I/O Error Message
90D8	Basic 'open'
90DF	Basic 'chrout'
90E5	Basic 'input'
90EB	Redirect Output
90FD	Redirect Input
9112	Perform [save]
9129	Perform [verify]
912C	Perform [load]
918D	Perform [open]
919A	Perform [close]
91AE	Get Load/Save Parameters
91DD	Get Next Byte Value
91E3	Get Character or Abort
91EB	Move to Next Parameter
91F6	Get Open/Close Params
9243	Release I/O String
9251	Call 'status'
9257	Call 'setlfs'
925D	Call 'setnam'
9263	Call 'getin'
9269	Call 'chrout'
926F	Call 'clrchn'
9275	Call 'close'
927B	Call 'clall'
9281	Print Following Text
9287	Set Load/Save Bank
928D	Call 'plot'
9293	Call 'get'
9299	Make Room For String
92EA	Garbage Collection
9409	Evaluate <cos>
9410	Evaluate <sin>
9459	Evaluate <tan>
9485	Trig Series
94B3	Evaluate <atn>
94E3	Series
9520	Print Using
99C1	Evaluate <instr>
9B0C	Evaluate <rdot>
9B30	Draw Line
9BFB	Plot Pixel
9C49	Examine Pixel
9C70	Set Hi-Res Color Cell
9CCA	Video Matrix Lines Hi
9CE3	Position Pixel
9D1C	Bit Masks
9D24	Calc Hi-Res Row/Column
9DF2	Restore Pixel Cursor
9E2F	Parse Graphics Command
9E32	Get Color Source Param
9F29	Conv Words Hi
9F3D	Conv Words Lo
A022	Move Basic to $1C01
A07E	Perform [catalog/directory]
A11D	Perform [dopen]
A134	Perform [append]
A157	Find Spare SA
A16F	Perform [dclose]
A18C	Perform [dsave]
A1A4	Perform [dverify]
A1A7	Perform [dload]
A1C8	Perform [bsave]
A218	Perform [bload]
A267	Perform [header]
A2A1	Perform [scratch]
A2D7	Perform [record]
A322	Perform [dclear]
A32F	Perform [collect]
A346	Perform [copy]
A362	Perform [concat]
A36E	Perform [rename]
A37C	Perform [backup]
A3BF	Parse DOS Commands
A5E7	Print 'missing file name'
A5EA	Print 'illegal device number'
A5ED	Print 'string too long'
A627	DOS Command Masks
A7E1	Print 'are you sure?'
A80D	Release String
A845	Set Bank 15
A84D	IRQ Work
AA1F	Perform [stash]
AA24	Perform [fetch]
AA29	Perform [swap]
AE64	Encrypted Message
AF00	Basic Vectors
B000	Perform [monitor]
B009	Break Entry
B00C	Print 'break'
B021	Print 'call' entry
B03D	Print 'monitor'
B050	Perform [r]
B053	Print 'pc sr. . .'
B08B	Get Command
B0BC	Error
B0BF	Print '?'
B0E3	Perform [x]
B0E6	Commands
B0FC	Vectors
B11A	Read Banked Memory
B12A	Write Banked Memory
B13D	Compare Banked Memory
B152	Perform [m]
B194	Perform [:]
B1AB	Perform [>]
B1CC	Print 'esc-o, up'
B1D6	Perform [g]
B1DF	Perform [j]
B1E8	Display Memory
B20E	Print ':<rvs-on>'
B231	Perform [c]
B234	Perform [t]
B2C3	Add 1 to Op 3
B2C6	Do Next Address
B2CE	Perform [h]
B337	Perform [lsv]
B3C7	Print 'error'
B3DB	Perform [f]
B406	Perform [a.]
B536	Print 'space <esc-q>'
B57C	Check 2 A-Matches
B57F	Check A-Match
B58B	Try Next Op Code
B599	Perform [d]
B5B1	Print '<cr> <esc-q>'
B5D4	Display Instruction
B5F5	Print '<3 spaces>'
B659	Classify Op Code
B6A1	Get Mnemonic Char
B6C3	Mode Tables
B715	Mode Characters
B721	Compacted Mnemonics
B7A5	Input Parameter
B7CE	Read Value
B88A	Number Bases
B88E	Base Bits
B892	Display 5-Digit Address
B8A5	Display 2-Digit Byte
B8A8	Print Space
B8AD	Print Cursor-Up
B8B4	New Line
B8B9	Blank New Line
B8C2	Output 2-Digit Byte
B8D2	Byte to 2 Ascii
B8E7	Get Input Char
B8E9	Get Character
B901	Copy Add0 to Add2
B90E	Calculate Add2-Add0
B922	Subtract
B93C	Subtract 1
B950	Increment Pointer
B960	Decrement Pointer
B974	Copy to Register Area
B983	Calculate Step/Range
B9B1	Perform [$+&%]
BA07	Convert o Decimal
BA47	Transfer Address
BA5D	Output Address
BA90	Perform [@]
C000	-cint-
C006	Get From Keyboard
C009	Screen Input Link
C00C	Screen Print Link
C00F	-screen-
C012	-scnkey-
C018	-plot-
C021	Define FN Key
C024	IRQ Link
C027	Upload 80 Col
C02A	Swap 40/80
C02D	Set Window
C033	Screen Address Low
C04C	Screen Address High
C065	I/O Link Vectors
C06F	Keyboard Shift Vectors
C07B	Initialize Screen
C142	Reset Window
C150	Home Cursor
C156	Goto Left Border
C15C	Set Up New Line
C17C	Do Screen Color
C194	(IRQ) Split Screen
C234	Get a Key
C29B	Input From Screen
C2BC	Read Screen Char
C2FF	Check For Quotes
C30C	Wrap Up Screen Print
C320	Ascii to Screen Code
C33E	Check Cursor Range
C363	Do New Line
C37C	Insert a Line
C3A6	Scroll Screen
C3DC	Delete a Line
C40D	Move Screen Line
C4A5	Clear a Line
C53C	Set 80-column Counter to 1
C53E	Set 80-column Counter
C55D	Keyboard Scan Subrtn
C651	Key Pickup & Repeat
C6DD	Programmed Keys
C6E7	Flash 40 Column Cursor
C72D	Print to Screen
C77D	Esc-o (escape)
C79A	Vectors
C7B6	Print Control Char
C802	Print Hi-Bit Char
C854	Chr$(29) Cursor Right
C85A	Chr$(17) Cursor Down
C875	Chr$(157) Cursor left
C880	Chr$(14) Text
C8A6	Chr$(11) Lock
C8AC	Chr$(12) Unlock
C8B3	Chr$(19) Home
C8BF	Chr$(146) Clear Rvs Mode
C8C2	Chr$(18) Reverse
C8C7	Chr$(2) Underline-On
C8CE	Chr$(130) Underline-Off
C8D5	Chr$(15) Flash-On
C8DC	Chr$(143) Flash-Off
C8E3	Open Screen Space
C91B	Chr$(20) Delete
C932	Restore Cursor
C94F	Chr$(9) Tab
C961	Chr$(24) Tab Toggle
C96C	Find Tab Column
C980	Esc-z Clear All Tabs
C983	Esc-y Set Default Tabs
C98E	Chr$(7) Bell
C9B1	Chr$(10) Linefeed
C9BE	Analyze Esc Sequence
C9DE	Vectors
CA14	Esc-t Top
CA16	Esc-b Bottom
CA1B	Set Window Part
CA24	Exit Window
CA3D	Esc-i Insert Line
CA52	Esc-d Delete Line
CA76	Esc-q Erase End
CA8B	Esc-p Erase Begin
CA9F	Esc-@ Clr Remainder of Scrn
CABC	Esc-v Scroll Up
CACA	Esc-w Scroll Down
CAE2	Esc-l Scroll On
CAE5	Esc-m Scroll Off
CAEA	Esc-c Cancel Auto Insert
CAED	Esc-a Auto Insert
CAF2	Esc-s Block Cursor
CAFE	Esc-u Underline Cursor
CB0B	Esc-e Cursor Non Flash
CB21	Esc-f Cursor Flash
CB37	Esc-g Bell Enable
CB3A	Esc-h Bell Disable
CB3F	Esc-r Screen Reverse
CB48	Esc-n Screen Normal
CB52	Esc-k End-of-Line
CB58	Get Screen Char/Color
CB74	Check Screen Line of Lo
CB81	Extend/Trim Screen Line
CB9F	Set Up Line Masks
CBB1	Esc-j Start-of-Line
CBC3	Find End-of-Line
CBED	Move Cursor Right
CC00	Move Cursor Left
CC1E	Save Cursor
CC27	Print Space
CC2F	Print Character
CC32	Print Fill Color
CC34	Put Char to Screen
CC5B	Get Rows/Columns
CC6A	Read/Set Cursor
CCA2	Define Function Key
CD2C	Esc-x Switch 40/80
CD57	Position 80-col Cursor
CD6F	Set Screen Color
CD9F	Turn Cursor On
CDCA	Set CRTC Register 31
CDCC	Set CRTC Register
CDD8	Read CRTC Register 31
CDDA	Read CRTC Register
CDE6	Set CRTC to Screen Address
CDF9	Set CRTC to Color Address
CE0C	Set Up 80 Column Char Set
CE4C	Ascii Color Codes
CE5C	System Color Codes
CE6C	Bit Masks
CE74	40-Col Init Values ($E0)
CE8E	80-Col Init Values ($0A40)
CEA8	Prog Key Lengths
CEB2	Prog Key Definitions
E000	Reset Code
E04B	MMU Set Up Bytes
E056	-restor-
E05B	-vector-
E073	Vectors to $0314
E093	-ramtas-
E0CD	Code For High RAM Banks

E105	RAM Bank Masks
E109	-ioinit-
E1DC	Set Up CRTC Registers
E1F0	Check Special Reset
E242	Reset to 64/128
E24B	Switch to 64 Mode
E263	Code to $02
E26B	Scan All ROMs
E2BC	ROM Addresses Hi
E2C0	ROM Banks
E2C4	Print 'cbm' Mask
E2C7	VIC 8564 Set Up
E2F8	CRTC 8563 Set Up Pairs
E33B	-talk-
E33E	-listen-
E43E	-acptr-
E4D2	-second-
E4E0	-tksa-
E503	-ciout- Print Serial
E515	-untlk-
E526	-unlsn-
E535	Reset ATN
E545	Set Clock High
E54E	Set Clock Low
E557	Set Data High
E560	Set Data Low
E569	Read Serial Lines
E573	Stabilize Timing
E59F	Restore Timing
E5BC	Prepare For Response
E5C3	Fast Disk Off
E5D6	Fast Disk On
E5FB	Fast Disk On/Off
E5FF	(NMI) Transmit RS-232
E64A	RS-232 Handshake
E68E	Set RS-232 Bit Count
E69D	(NMI) RS-232 Receive
E75F	Send to RS-232
E795	Connect RS-232 Input
E7CE	Get From RS-232
E7EC	Interlock RS-232/Serial
E805	(NMI) RS-232 Control I/O
E850	RS-232 Timings
E878	(NMI) RS-232 Receive Timing
E8A9	(NMI) RS-232 Transmit Timing
E8D0	Find Any Tape Header
E919	Write Tape Header
E980	Get Buffer Address
E987	Get Tape Buffer Start & End Addrs
E99A	Find Specific Header
E9BE	Bump Tape Pointer
E9C8	Print 'press play . . .'
E9DF	Check Tape status
E9E9	Print 'press record..'
E9F2	Initiate Tape Read
EA15	Initiate Tape Write
EA26	Common Tape Code
EA7D	Wait For Tape
EA8F	Check Tape Stop
EAA1	Set Read Timing
EAEB	(IRQ) Read Tape Bits
EC1F	Store Tape Chars
ED51	Reset Pointer
ED5A	New Char Set Up
ED69	Send Transitn to Tape
ED8B	Write Data to Tape
ED90	(IRQ) Tape Write
EE2E	(IRQ) Tape Leader
EE57	Wind Up Tape I/O
EE9B	Switch IRQ Vector
EEA8	IRQ Vectors
EEB0	Kill Tape Motor
EEB7	Check End Address
EEC1	Bump Address
EEC8	(IRQ) Clear Break
EED0	Control Tape Motor
EEEB	-getin-
EF06	-chrin-
EF48	Get Char From Tape
EF79	-chrout-
EFBD	-open-
F0B0	Set CIA to RS-232
F0CB	Check Serial Open
F106	-chkin-
F14C	-chkout-
F188	-close-
F1E4	Delete File
F202	Search For File
F212	Set File Parameters
F222	-clall-
F226	-clrchn-
F23D	Clear I/O Path
F265	-load-
F27B	Serial Load
F32A	Tape Load
F3A1	Disk Load
F3EA	Burst Load
F48C	Close Off Serial
F4BA	Get Serial Byte
F4C5	Receive Serial Byte
F503	Toggle Clock Line
F50C	Print 'u0' Disk Reset
F50F	Print 'searching'
F521	Send File Name
F533	Print 'loading'
F53E	-save-
F5B5	Terminate Serial Input
F5BC	Print 'saving'
F5C8	Save to Tape
F5F8	-udtim-
F63D	Watch For RUN or Shift
F65E	-rdtim-
F665	-settim-
F66E	-stop-
F67C	Print 'too many files'
F67F	Print 'file open'
F682	Print 'file not open'
F685	Print 'file not found'
F688	Print 'device not present'
F68B	Print 'not input file'
F68E	Print 'not output file'
F691	Print 'missing file name'
F694	Print 'illegal device no'
F697	Error #0
F6B0	Messages
F71E	Print If Direct
F722	Print I/O Message
F731	-setnam-
F738	-setlfs-
F73F	Set Load/Save Bank
F744	-rdst-
F757	Set Status Bit
F75C	-setmsg-
F75F	Set Serial Timeout
F763	-memtop-
F772	-membot-
F781	-iobase-
F786	Search For SA
F79D	Search & Set Up File
F7A5	Trigger DMA
F7AE	Get Char From Memory
F7BC	Store Loaded Byte
F7C9	Read Byte to be Saved
F7D0	Get Char From Memory Bank
F7DA	Store Char to Memory Bank
F7E3	Compare Char With Memory Bank
F7EC	Load Mem Control Mask
F7F0	Bank Masks
F800	Subrtns to $02A2-$02FB
F85A	DMA Code to $03F0
F867	Check Auto Start ROM
F890	Check For Boot Disk
F90B	Print 'booting'
F92F	Print '. . .'
F98B	Wind Up Disk Boot
F9B3	Read Next Boot Block
F9FB	To 2-Digit Decimal
FA08	Block Read
FA15	Print '#i'
FA17	Print a Message
FA40	NMI Sequence
FA65	(IRQ) Normal Entry
FA80	Keyboard Matrix Un-Shifted
FAD9	Keyboard Matrix Shifted
FB32	Keyboard Matrix C-Key
FB8B	Keyboard Matrix Control
FBE4	Keyboard Matrix Caps Lock
FF00	MMU Controls
FF05	NMI Transfer Entry
FF17	IRQ Transfer Entry
FF33	Return From Interrupt
FF3D	Reset Transfer Entry
FF47	Jumbo Jump Table
FFFA	Transfer Vectors

8502 Processor I/O Registers

0000	X	0 = in	1 = out	0 = in	1 = out	1 = out	1 = out	1 = out	00000
0001	X	Caps Key	Tape Motor	Tape Sense	Tape Output	HiRes	LoRes	Color Access	00001

8722 Memory Management Unit

D500	RAM select 0–3		HIGH RAM /ROM		MID RAM /ROM		LO RAM	C GEN	54528
D501–D504	Preconfiguration registers; Similar to D500, above								54529–54532
D505	40/80 Key	C64 Mode	Cartr–Sense Color–Bank		Fast Disk	X	X	Z80	54533
D506	Video–Bank		X	X	Shared RAM hi	Shared RAM low	Shared RAM 0 = 1K		54534
D507	Zero Page Pointer ($0000) L								54535
D508	Zero Page Pointer ($0000) H								54536
D509	Stack Page Pointer ($0000) L								54537
D50A	Stack Page Pointer ($0000) H								54538

DMA Controller

DF00	Busy	Fault	X	X	X	X	X	X	57088
DF01	Exec	Sum	X	X	IRQ	Inc	Mode		57089
DF02	Host Address L								57090
DF03	Host Address H								57091
DF04	Expansion Address L								57092
DF05	Expansion Address H								57093
DF06	X	X	X	X	X	Expansion Bank			57094
DF07	Transfer Length L								57095
DF08	Transfer Length H								57096
DF09	Checksum								57097
DF0A	Version, Maximum–Memory								57098

6526 CIA 1 (IRQ)

(Same as CIA 1 for C64, until DC0C)

DC00	Paddle Select A	Paddle Select B		Joystick 0 Fire	Right	Left	Down	Up	PRA	56320
	Keyboard Row Select (inverted)									
DC01				Joystick 1 Fire	Right	Left	Down	Up	PRB	56321
	Keyboard Column Read									
DC02	$FF - All Output								DDRA	56322
DC03	$00 - All Input								DDRB	56323
DC04	Timer A L								TAL	56324
DC05	Timer A H								TAH	56325
DC06	Timer B L								TBL	56326
DC07	Timer B H								TBH	56327
DC0C	Serial (shift) Register									56332
DC0D	IRQ	X	X	Flag	S.Reg	X	Tim.B	Tim.A		56333
DC0E	S Reg I/O			Load	O/S	Timer A Toggle		Start		56334
DC0F				Load	O/S	Timer B		Start		56335

6526 CIA 2 (NMI)

(Same as CIA 2 for C64)

DD00	Serial IN	Clock IN	Serial OUT	Clock OUT	ATN OUT	RS232 OUT	Video	Block	PRA	56576
DD01	DSR IN	CTS IN		DCD* IN	RI* IN	DTR OUT	RTS OUT	RS232 IN	PRB**	56577
DD02	IN	IN	OUT	OUT	OUT	OUT	OUT	OUT	DDRA	56578
DD03	$06 for RS232								DDRB	56579
DD04	Timer A L								TAL	56580
DD05	Timer A H								TAH	56581
DD06	Timer B L								TBL	56582
DD07	Timer B H								TBH	56583
DD0D				RS232 IN			Timer B	Timer A	ICR	56589
DD0E								Timer A Start	CRA	56590
DD0F								Timer B Start	CRB	56591

* Connected but not used by O.S.

** PRB is the Parallel User Port

DDRA = $3F at reset

8564 Video Chip
Control & Miscellaneous Registers

Address	Fields					Decimal
D011		Extended Clr. Mode	Bit Map	Display Enable	Row Select / Y-Scroll	53265
D012	Raster Register					53266
D013	Light Pen Input X					53267
D014	Light Pen Input Y					53268

Address	Fields					Decimal
D016	x	x	Reset	Multi Colour	Column Select / X-Scroll	53270

Address	Fields								Decimal
D018	Screen: VM13	VM12	VM11	VM10	Character Base: CB13	CB12	CB11	x	53272
D019	IRQ			Interrupt Sense:	Light Pen	Spr-Spr Collision	Spr-Back Collision	Raster	53273
D01A				Interrupt Enable:	Light Pen	Spr-Spr Collisions	Spr-Back Collisions	Raster	53274

Colour Registers

Address			Decimal
D020	X	Exterior Colour (Border)	53280
D021	X	Background Colour #0	53281
D022	X	Background Colour #1	53282
D023	X	Background Colour #2	53283
D024	X	Background Colour #3	53284
D025	X	Sprite MultiColour #0	53285
D026	X	Sprite MultiColour #1	53286

Address									Decimal
D02F	x	x	x	x	x	[Keyboard Rows]			53295
D030	X	X	X	X	X	X	Test	Fast Clock	53296

8564 Video Chip
Sprite Registers

Sprite 0	Sprite 7		Sprite 0	Sprite 7
D000	D00E	X Position	53248	53262
D001	D00F	Y Position	53249	53263
D027	D02E	Sprite Colour	53287	53294

Bit For Sprite#: 7 6 5 4 3 2 1 0

Address		Decimal
D010	X-Position High	53264
D015	Sprite Enable Flags	53269
D017	Y-Expand	53271
D01B	Background Priority	53275
D01C	Sprite MultiColour Mode	53276
D01D	X-Expand	53277
D01E	Interrupt: Sprite Collision	53278
D01F	Interrupt: Background Collision	53279

6581 SID Sound Chip
(Identical to 6581 on C64)

Voice 1	Voice 2	Voice 3		Voice 1	Voice 2	Voice 3
D400	D407	D40E	Frequency L	54272	54279	54286
D401	D408	D40F	Frequency H	54273	54280	54287
D402	D409	D410	Pulse Width L	54274	54281	54288
D403	D40A	D411	0 0 0 0 / Pulse Width H	54275	54282	54289
D404	D40B	D412	Voice Type: NSE, PUL, SAW, TRI / Key	54276	54283	54290
D405	D40C	D413	Attack Time: 2ms-8sec / Decay Time: 6ms-24sec	54277	54284	54291
D406	D40D	D414	Sustain Level: / Release Time: 6ms-24sec	54278	54285	54292

Voices are "write-only"

Address									Decimal
D415	0	0	0	0	0			L	54293
D416	Filter Frequency H								54292
D417	Resonance				Ext	Filter Voices: V3	V2	V1	·54295
D418	V3 off	Passband: HI	BP	LO	Master Volume				54296

Filter and Volume (write only)

Address		Decimal
D419	Paddle X (A/D #1)	54297
D41A	Paddle Y (A/D #2)	54298
D41B	Noise 3 (random)	54299
D41C	Envelope 3	54300

Sense (read only)

Note: Special Voice Features
(TEST, RING, MOD, SYNC)
are omitted from the above diagram

8563 80-Column CRT Controller

D600 read (status):

D600	Status	Light Pen	Vert Blank	X	X	X	X	X	54784

D600 54784	D601 54785	Typical Value
0 $00	Horizontal Total	126
1 $01	Horizontal Characters Displayed (80)	80
2 $02	Horizontal Sync position	102
3 $03	Vertical Sync Width / Horizontal Sync Width	1 and 3
4 $04	X / Vertical Total	32 or 39
5 $05	X X X / Vertical Total Adjust	0
6 $06	X / Vertical Displayed (25)	25
7 $07	X / Vertical Sync Position	29 or 32
8 $08	X X X X X X / Interlace	0
9 $09	X X X / Scan Lines per Character	7
10 $0A	X / Cursor Mode / Cursor Start	32
11 $0B	X X X / Cursor End	7
12 $0C	X X / Display Address H	0
13 $0D	Display Address L	0
14 $0E	Cursor Address H	0
15 $0F	Cursor Address L	0
16 $10	Light Pen Input H	varies
17 $11	Light Pen Input L	varies
18 $12	Video RAM Address (See register 31) H	varies
19 $13	Video RAM Address (See register 31) L	varies
20 $14	Colour Address H	8
21 $15	Colour Address L	0
22 $16	Character Total / Character Display Horizontal	120
23 $17	X X X / Character Display Vertical	8
24 $18	Block Copy / Scrn RVS / Blink Rate / V Scroll	32
25 $19	Bit Map / Colour Enable / Semi Graph / Wide Pixel / H Scroll	64 or 71
26 $1A	Foreground Colour / Background Colour	240
27 $1B	Scroll Control Horizontal	0
28 $1C	Char Set Address / RAM / X X X X	32
29 $1D	X X X / Underline Scan Line Count	7
30 $1E	Character Count	varies
31 $1F	Video RAM data (see registers 18,19)	varies
32 $20	Block Copy Start Address H	varies
33 $21	Block Copy Start Address L	varies
34 $22	Display Enable begin	125
35 $23	Display Enable end	100
36 $24	X X X X / DRAM Refresh Rate	5